DEVELOPMENT OF A PERSONAL COMPUTER BASED APPROACH TO AIRCRAFT PARAMETER IDENTIFICATION

THESIS

Mark S. Erickson
Captain, USAF

AFIT/GAE/ENY/91M-1

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/91M-1

DEVELOPMENT OF A PERSONAL COMPUTER BASED APPROACH TO AIRCRAFT PARAMETER IDENTIFICATION

THESIS

Mark S. Erickson
Captain, USAF

AFIT/GAE/ENY/91M-1

Approved for public release; distribution unlimited

AFIT/GAE/ENY/91M-1

DEVELOPMENT OF A PERSONAL COMPUTER BASED APPROACH TO AIRCRAFT PARAMETER IDENTIFICATION

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Aerospace Engineering

Mark S. Erickson, B.S.
Captain, USAF

March 1991

March 1991 M.S. Thesis

Development of a Personal Computer Based Approach to Aircraft Parameter Identification

Mark S. Erickson, B.S., Captain, USAF

School of Engineering
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

AFIT/GAE/ENY/91M-1

USAF Test Pilot School
Edwards AFB, California 93523

Approved for public release;
distribution unlimited

This thesis develops and tests a personal computer based approach to the aircraft parameter identification problem. The approach was tested analytically at the Air Force Institute of Technology, Wright-Patterson AFB, Ohio. These tests were conducted using simplified short period approximation and second-order transfer function models of longitudinal aircraft response. Initial evaluations were accomplished using simulated, noise-free data.

The longitudinal aircraft model was modified prior to flight test, for both convenience and necessity. This more complex model was evaluated during a flight test program at the USAFTPS, Edwards AFB, California. A T-38A aircraft was used as a tool to demonstrate and validate the process. Stability derivative estimates from the personal computer based approach were compared with those from the existing mainframe MMLE3 program available at the Air Force Flight Test Center. Second-order system parameters calculated using these derivatives were also compared to those obtained from frequency response analysis routines available at the AFFTC.

Thesis Advisor: Major (Dr.) Daniel Gleason
Adjunct Assistant Professor of Aerospace Engineering

Maximum Likelihood Estimation, 133
Parameter Identification

UNCLASSIFIED UNCLASSIFIED UNCLASSIFIED UL

Preface

This thesis presents the results of the development and evaluation of an aircraft parameter identification capability for the United States Air Force Test Pilot School (USAFTPS) at Edwards Air Force Base, California. It represents research effort at the Air Force Institute of Technology (AFIT), Wright-Patterson Air Force Base, Ohio, and a flight test program at the USAFTPS.

I would like to thank the faculty and staff members of both the Air Force Institute of Technology and the Test Pilot School, who helped me along the way. In particular, Dr. Robert Calico and Major (Dr.) Daniel Gleason.

I also want to express my gratitude to the Have Stability test team of USAFTPS Class 90A. The team consisted of Captains Jeff Borton, Tom Masiello, Yoshiteru Miwa (Japanese Air Self Defense Force), Henk-Jan Muusse (Royal Netherlands Air Force), and myself.

In addition, special thanks to Mr. Tom Twisdale and Mr. Mike Nelson at Edwards Air Force Base, whose technical expertise and hard work made the successful completion of the flight test portion possible.

Finally, I must thank my wife, Gretchen, for her patience and support throughout the AFIT/USAFTPS program.

Mark S. Erickson

Table of Contents

Table of Contents (Concluded)

List of Figures

List of Figures (Continued)

List of Figures (Concluded)

List of Tables

List of Abbreviations and Symbols

Abbreviation	Definition
AFB	Air Force Base
AFFTC	Air Force Flight Test Center
AFIT	Air Force Institute of Technology
ASCII	American standard code for information interchange
CG	center of gravity (percent chord)
dB	decibels
deg	degrees
FRA	frequency response analysis
ft	feet
FTP	file transfer protocol
in	inches
lb, lbs	pounds
LOES	lower-order equivalent system
LONG.	longitudinal
MAC	mean aerodynamic chord (feet)
MMLE	Modified Maximum Likelihood Estimation
MMLE3	Fortran MMLE program
PC	personal computer
psf	pounds per square foot
P2SS	parameter-to-state-space
rad	radians
RMCC	Ridley Mission Control Center

List of Abbreviations and Symbols (continued)

Abbreviation	Definition
s, sec	seconds
S/N	serial number
TPS	Test Pilot School
USAF	United States Air Force
USAFTPS	United States Air Force Test Pilot School
YAPS	Yaw, Angle of Attack, Pitot-static System

Symbol	Definition
A	continuous system dynamics matrix
B	control distribution matrix
c	chord length, feet
C	state measurement matrix
C_M	pitching moment coefficient
C_{Mo}	pitching moment coefficient bias
C_{Mq}	$\partial C_M/\partial(qc/2V)$, per rad, per deg
$C_{M\alpha}$	$\partial C_M/\partial\alpha$, per rad, per deg
$C_{M\delta e}$	$\partial C_M/\partial\delta e$, per rad, per deg
C_N	total normal force coefficient
C_{No}	normal force coefficient bias
$C_{N\alpha}$	$\partial C_N/\partial\alpha$, per rad, per deg
$C_{N\delta e}$	$\partial C_N/\partial\delta e$, per rad, per deg

List of Abbreviations and Symbols (continued)

Symbol	Definition
D	discretized transmission matrix
E { }	expected value
F	square root of state noise spectral density
g	acceleration of gravity, 32.17 ft/sec^2
G	square root of measurement noise covariance matrix
h	pressure altitude, feet
I_{xx}	moment of inertia about the x-axis, slug-ft^2
I_{xy}	product of inertia about the x- and y-axes, slug-ft^2
I_{xz}	product of inertia about the x- and z-axes, slug-ft^2
I_{yy}	moment of inertia about the y-axis, slug-ft^2
I_{yz}	product of inertia about the y- and z-axes, slug-ft^2
I_{zz}	moment of inertia about the z-axis, slug-ft^2
J	cost function
K_α	flow amplification factor for α
K_θ	q/δe transfer function gain
m	mass, slugs
M_q	dimensional variation of pitching moment with pitch rate, per sec

Symbol	Definition
M_{Tu}	dimensional variation of pitching moment due to thrust with speed, per (ft-sec)
M_u	dimensional variation of pitching moment with speed, per (ft-sec)
$M_{T\alpha}$	dimensional variation of pitching moment due to thrust with α, per sec^2
M_α	dimensional variation of pitching moment with α, per sec^2
$M_{\dot{\alpha}}$	dimensional variation of pitching moment with $\dot{\alpha}$, per sec
$M_{\delta e}$	dimensional variation of pitching moment with α, per sec^2
n	state noise vector
N_Z	normal load factor, g
p	roll rate, rad/sec, deg/sec
$p(z\|\zeta)$	likelihood functional
q	pitch rate, rad/sec, deg/sec
$\bar{q}$	dynamic pressure, lb/ft^2
Q	state noise matrix
r	yaw rate, rad/sec, deg/sec
S	reference area, ft^2
u	input vector or forward velocity perturbation, ft/sec
U_1	steady state forward velocity, ft/sec

List of Abbreviations and Symbols (continued)

Symbol	Definition
V	true velocity, ft/sec
W	a-priori weighting matrix
x	state vector or longitudinal axis designation
X_{An}	x distance (+ forward) from CG to normal accelerometer, feet
X_{Tu}	dimensional variation of longitudinal force due to thrust (stability axis) with speed, per sec
X_u	dimensional variation of longitudinal force (stability axis) with speed, per sec
X_α	dimensional variation of longitudinal force (stability axis) with α, ft/sec^2
$X_{\alpha b}$	x distance (+ forward) from CG to angle of attack vane, feet
$X_{\delta e}$	dimensional variation of longitudinal force (stability axis) with δe, ft/sec^2
y	output vector or lateral axis designation
Y_{An}	y distance (+ right) from CG to normal accelerometer, feet
$Y_{\alpha b}$	y distance (+ right) from CG to angle of attack vane, feet
z	measured observation vector or directional axis designation
$\tilde{z}$	predicted estimate of z
Z_{An}	z distance (+ down) from CG to normal accelerometer, feet

List of Abbreviations and Symbols (continued)

Symbol	Definition
Z_q	dimensional variation of z-axis force (stability axis) with pitch rate, ft/sec
Z_u	dimensional variation of z-axis force (stability axis) with speed, per sec
Z_α	dimensional variation of z-axis force (stability axis) with α, ft/sec^2
$Z_{\dot{\alpha}}$	dimensional variation of z-axis force (stability axis) with $\dot{\alpha}$, ft/sec
$Z_{\delta e}$	dimensional variation of z-axis force (stability axis) with δe, ft/sec^2
$1/T_{\theta 2}$	q/δe transfer function numerator term
α	angle of attack, rad, deg
β	angle of sideslip, rad, deg
Γ	discrete control distribution matrix
δe	elevator deflection, rad, deg
ζ	parameter vector
ζ_o	initial guess of parameter vector
ζ_{sp}	short period damping ratio
η	measurement noise vector
θ	pitch angle, rad, deg
$\dot{\theta}_o$	$\dot{\theta}$ bias, rad/sec, deg/sec
Σ	summation
ϕ	bank angle, rad, deg
Φ	state transition matrix

List of Abbreviations and Symbols (concluded)

Symbol	Definition
ω_{sp}	short period natural frequency, rad/sec
∇	gradient
∇^2	second gradient
∂	partial derivative
%	percent
*	transpose

General Notation	Definition
m subscript	measurement
1 subscript	steady state value
· above	denotes derivative with respect to time
·· above	denotes second derivative with respect to time

Abstract

Parameter estimation is a key component of many flight test programs, particularly during the initial development of new designs. Accurate estimates of key aerodynamic derivatives are required to update wind tunnel models and simulations. These estimates facilitate envelope expansion flights, flight control system design, and increase fidelity of ground simulators. With the advent of more highly augmented aircraft, more sophisticated techniques are required (over and above conventional analysis) to provide accurate estimates of aerodynamic derivatives. While there are computer programs which have been used for years at the Air Force Flight Test Center (AFFTC), such as Modified Maximum Likelihood Estimation (MMLE3), they are mainframe computer based, and require considerable time to set up for a particular test program.

This thesis develops and tests a personal computer based approach to the aircraft parameter identification problem. The approach was tested analytically at the Air Force Institute of Technology, Wright-Patterson AFB, Ohio. These tests were conducted using simplified short period approximation and second-order transfer function models of longitudinal aircraft response. Initial evaluations were accomplished using simulated, noise-free data.

The longitudinal aircraft model was modified prior to flight test, for both convenience and necessity. This more complex model was evaluated during a flight test program at the USAFTPS, Edwards AFB, California. A T-38A aircraft was used as a tool to demonstrate and validate the process. Stability derivative estimates from the personal computer based approach were compared with those from the mainframe MMLE3 program. Second-order system parameters calculated using these derivatives were also compared to those obtained from frequency response analysis routines also available on the Cyber computer at the AFFTC.

DEVELOPMENT OF A PERSONAL COMPUTER BASED APPROACH TO AIRCRAFT PARAMETER IDENTIFICATION

I. Introduction

Background

Parameter estimation is a key component of many flight test programs, particularly during the initial development of new designs. Accurate estimates of key aerodynamic derivatives are required to update wind tunnel models and simulations. These estimates facilitate envelope expansion flights, flight control system design, and increase fidelity of ground simulators. With the advent of more highly augmented aircraft, more sophisticated techniques are required (over and above conventional analysis) to provide accurate estimates of aerodynamic derivatives.

Conventional dynamic techniques consist of maneuvers such as stick and rudder pedal pulses or doublets. The time history responses of the aircraft are analyzed using log decrement or time ratio methods to determine damping ratio and natural frequency (1:8.47-8.53). In general, these conventional techniques have three primary problem areas: accuracy, ease, and speed of testing and analysis. Conventional testing requires specific maneuvers for each

parameter estimate. In some cases, highly augmented aircraft have control systems which prevent analysis of basic aircraft motion. Even in those cases where basic aircraft motion can be analyzed, the process is time consuming and often inaccurate. Finally, from the standpoint of flight safety, highly augmented control systems could mask serious handling qualities problems, making it difficult (and perhaps dangerous) to extrapolate handling qualities trends (11:11).

Derivative analysis techniques offer several advantages over conventional techniques. First of all, derivatives provide better insight into handling qualities discrepancies between predicted and actual flight test data. Because of the dependence of many conventional flight test relationships on multiple stability derivatives, it is difficult to determine which derivative is in error if a discrepancy occurs. Derivative analysis also allows for extrapolation and standardization of data. Extrapolation allows for a safer approach to envelope expansion and high angle of attack testing. Standardization makes it possible to make direct comparisons of data from different tests or different aircraft. Techniques for derivative analysis can be easily managed in computers, and the results obtained can be used to update simulators with flight test data. Finally, there is an overall savings in flight test time.

This is becoming especially important as costs associated with flight test increase (11:19-21).

Objective

The lack of a flexible implementation of parameter estimation has been a stumbling block in the past. While there are computer programs which have been used for years at the Air Force Flight Test Center (AFFTC), such as Modified Maximum Likelihood Estimation (MMLE3), they are mainframe computer based, and require considerable time to set up for a particular test program. Recently, a commercially available personal computer software package has become available which implements the MMLE3 problem formulation within the PC Matlab program. PC Matlab is a matrix operation program which can be used to solve practical engineering and mathematical problems. PC Matlab is designed for specific applications such as control theory and signal processing. (8:3). Two additional "toolboxes" are used for this project: the Control System Toolbox (6) and the State Space Identification Tool (10). The State Space Identification Tool implements the MMLE3 formulation, but it is written in terms of general system identification, and is not designed for specific application to aircraft equations of motion. The objective of this thesis is to modify and test the existing personal computer software package to accomplish aircraft parameter estimation. The

end goal is to verify and validate the results of the personal computer based MMLE by comparing flight test results with those from the mainframe MMLE3 program, as well as cross checking the results with conventional analysis techniques and frequency response analysis methods.

Section II will discuss the theory behind parameter estimation, including maximum likelihood estimation and some of the particular capabilities of the existing Fortran program, MMLE3. Section III presents the work done using simulated data to check out the PC Matlab MMLE software. Section IV summarizes the final preparations which were necessary prior to proceeding with flight test. Section V describes the details of the flight test program, which was the ultimate test of the personal computer based approach. Finally, section VI summarizes the conclusions drawn and recommendations made as a result of this effort.

II. Parameter Estimation Theory

The general problem of aircraft parameter estimation can be described in fairly simple terms. It is first assumed that a set of dynamic equations exists which accurately model the system in question. This model contains various unknowns, which are the parameters of interest. The system is then subjected to a predefined input, designed to produce a response which will allow estimation of these unknowns. Once the actual input and responses are measured, values for the unknown parameters are derived from the constraint that the model responses should match the actual responses. This whole process sounds fairly simple, but several problems arise when trying to apply the theory to a real system.

This first problem is measurement noise. It is basically impossible to measure a real system perfectly. Measurement noise makes it impossible to identify actual values of the parameters of interest. This causes the problem to be considered parameter estimation, rather than parameter identification.

The second problem is state noise. State noise is any outside (unmeasured) source that causes unexpected system response. For the aircraft problem, atmospheric turbulence is the most common form of state noise. With both

measurement and state noise, the problem becomes more difficult than with either of the two alone.

Finally, modeling error can corrupt the reliability of the parameter estimates. In basic terms, it is assumed throughout the parameter estimation process that there is a dynamic model which accurately represents the system response. In other words, for some set of "true" values for each unknown parameter, it is assumed that the model and actual responses match. In most cases, simplified dynamic models do not exactly describe the real system.

For purposes of aircraft parameter identification, the MMLE3 program has been used for many years as the standard at the AFFTC. The basics of the MMLE3 program lie in the theory of Maximum Likelihood Estimation, which is reviewed in the following subsection.

Maximum Likelihood Estimation

The Modified Maximum Likelihood Estimation (MMLE3) program is a Fortran computer program which is used as the standard for stability derivative extraction at the AFFTC. This subsection describes the method used by MMLE3 for maximum likelihood estimation, as developed by Maine and Iliff (7). In general, the parameter estimates are obtained by choosing a parameter vector ζ which maximizes the likelihood functional $p(z|\zeta)$, where z is the measured system response. The critical part of the development is the

actual definition of $p(z|\zeta)$, which leads directly to the cost function for the minimization process. With both state and measurement noise, the continuous system model is

$$\dot{x}(t) = Ax(t) + Bu(t) + Fn(t) \tag{1}$$

where

x = state vector

u = input vector

n = state noise vector

A = continuous system dynamics matrix

B = control distribution matrix

F = square root of the state noise spectral density

The discrete measurement equation is

$$z(t_i) = Cx(t_i) + Du(t_i) + G\eta(t_i) \tag{2}$$

where

z = measured observation vector

η = measurement noise vector

C = state measurement matrix

D = discretized transmission matrix

G = square root of the measurement noise covariance matrix

It is assumed that the state noise vector is zero mean, Gaussian, and white, with unit spectral density. Also, the measurement noise vectors are assumed to be zero mean, Gaussian, independent random vectors with identity covariance (7:11).

With this system model, the likelihood functional $p(z|\zeta)$ can be written directly. However, because of its simplicity, the logarithm of the likelihood ratio is usually used. For this model, the log likelihood function is

$$\log p(z|\zeta) = -\frac{1}{2}\sum_{i=1}^{N}[\tilde{z}_\zeta(t_i)-z(t_i)]^*(GG^*)^{-1}[\tilde{z}_\zeta(t_i)-z(t_i)] - \frac{N}{2}\log|GG^*| - \frac{N}{2}m\log 2\pi \qquad (3)$$

where $\tilde{z}_\zeta$ is a predicted estimate of z computed using a Kalman filter. The last term in equation (3) is a constant, and if G is known, the next to last term is also constant. Therefore they can be neglected in the maximization process (7:11-12).

If the log likelihood function is multiplied by -1, the problem becomes a minimization rather than a maximization. The resulting problem is to minimize the cost function

$$J(\zeta) = \frac{1}{2}\sum_{i=1}^{N}[\tilde{z}_\zeta(t_i)-z(t_i)]^*(GG^*)^{-1}[\tilde{z}_\zeta(t_i)-z(t_i)] \qquad (4)$$

Cost Function Minimization

Now that the cost function has been defined, the algorithm for the minimization needs to be specified. The Newton-Raphson technique is well known for iterative functional minimization. The Newton-Raphson algorithm requires an initial starting point for the estimated parameter vector, ζ. Subsequent values for ζ are determined from the equation

$$\zeta_{i+1} = \zeta_i - \left[\nabla_\zeta^2 J(\zeta_i)\right]^{-1} \nabla_\zeta^* J(\zeta_i) \tag{5}$$

The first and second gradients in equation (5) are defined as

$$\nabla_\zeta J(\zeta) = \sum_{i=1}^{N} \left[\tilde{z}_\zeta(t_i) - z(t_i)\right]^* (GG^*)^{-1} \nabla_\zeta \left[\tilde{z}_\zeta(t_i)\right] \tag{6}$$

and

$$\nabla_\zeta^2 J(\zeta) = \sum_{i=1}^{N} \nabla_\zeta^* \left[\tilde{z}_\zeta(t_i)\right]^* (GG^*)^{-1} \nabla_\zeta \left[\tilde{z}_\zeta(t_i)\right] + \sum_{i=1}^{N} \left[\tilde{z}_\zeta(t_i) - z(t_i)\right]^* (GG^*)^{-1} \nabla_\zeta^2 \left[\tilde{z}_\zeta(t_i)\right] \tag{7}$$

The second term in equation (7) involves significant computation for a realistic number of parameters. It should be small near the minimum, because $\tilde{z}$ - z should be small. Therefore, this term can be neglected if the process starts

close enough to the solution (7:14). This results in the second gradient being defined by

$$\nabla_\zeta^2 J(\zeta) = \sum_{i=1}^{N} \nabla_\zeta^*[\tilde{z}_\zeta(t_i)]^*(GG^*)^{-1}\nabla_\zeta[\tilde{z}_\zeta(t_i)] \qquad (8)$$

This approximation is called the Gauss-Newton approximation, and is also known as the modified Newton-Raphson or Newton-Balakrishnan technique. The Gauss-Newton approximation is much easier for computation, since it does not require calculation of the second gradient of the innovations. It can also expedite the convergence process (4:3).

A-Priori Information

In addition to the data available from processing one particular maneuver through the parameter estimation routine, other estimates may be available from prior runs at the same test case or from entirely different sources. The algorithm can be set up to account for this information by adding a penalty function to the cost function for deviations from the a-priori estimates. The quadratic penalty function, when added to the cost function yields a cost function of the form

$$J(\zeta) = \frac{1}{2}\sum_{i=1}^{N}[\hat{z}_\zeta(t_i) - z(t_i)]^*(GG^*)^{-1}[\hat{z}_\zeta(t_i) - z(t_i)]$$

$$+ \frac{1}{2}(\zeta-\zeta_0)^* W(\zeta-\zeta_0) \qquad (9)$$

where the W matrix is a weighting matrix corresponding to the a-priori values in the vector ζ_0 (7:15).

While the a-priori function can assist convergence in some cases, it should be used with caution since it also biases the estimates (possibly in the wrong direction). The a-priori weighting can be used only for initial convergence, then removed to arrive at the unbiased parameter estimates.

Cramér-Rao Bounds

Since the maximum likelihood estimation method yields just that ... estimates, some accuracy measure is necessary. The Cramér-Rao bounds are a good analytical measure of estimation accuracy (4:11). In many cases, the scatter of the estimates would be sufficient for determining accuracy. However, if there are not enough estimates available for a particular test point, even the scatter cannot be used. The Cramér-Rao bound is defined by

$$Variance(\zeta) \geq (E\{[\nabla_\zeta \log p(z|\zeta)]^*[\nabla_\zeta \log p(z|\zeta)]\})^{-1} \qquad (10)$$

It is important to note that the Cramér-Rao bounds are lower bounds for the variance. In other words, the variance is at

least as large as the Cramér-Rao bound on the estimate (7:16).

For the case where there is no state noise, the Cramér-Rao bound can be determined by

$$Variance(\zeta) \approx \left\{ \sum_{i=1}^{N} \left[\nabla_\zeta^* \bar{z}_\zeta(t_i)^* \right] (GG^*)^{-1} \left[\nabla_\zeta \bar{z}_\zeta(t_i) \right] \right\}^{-1} \qquad (11)$$

Equation (11) is the inverse of the Gauss-Newton approximation to the second gradient from equation (8). Because the second gradient calculation is already required, the Cramér-Rao bound calculation is minimal.

Residual Power Estimation

In most of the above discussion, it has been assumed that the G matrix is known. However, the G matrix can be estimated by maximizing the log likelihood function in equation (3). The problem is that $\bar{z}_\zeta$ is a function of G. There is a process used to estimate G which estimates it and the rest of the unknown parameters separately. Equation (3) is maximized with respect to G, without considering the effect of G on $\bar{z}_\zeta$. In particular, the estimator is

$$GG^* = \frac{1}{N} \sum_{i=1}^{N} \left[\bar{z}_\zeta(t_i) - z(t_i) \right] \left[\bar{z}_\zeta(t_i) - z(t_i) \right]^* \qquad (12)$$

The entire process is accomplished by first estimating the unknown parameters other than G. The G matrix is then estimated using $\tilde{z}_\zeta$ evaluated at the new ζ to get the new estimate of G. These steps are repeated until convergence (7:17).

Summary

The above section described the theory and algorithms behind maximum likelihood estimation. The State Space Identification Tool for the PC Matlab personal computer program implements the maximum likelihood algorithm. The task at hand is to apply the general theory to the specific application of aircraft equations of motion. The process of developing models and getting the software to work for aircraft applications using both simulated and flight test data is presented in subsequent sections.

III. Simulation

Introductory work was done using simulated data to check out the personal computer software for identifying aircraft response parameters. The primary areas of concern were: developing an adequate model for the short period aircraft response, generating test inputs and outputs, creating the macro files in PC Matlab to tailor the MMLE software to the aircraft model, and running the software with the simulated data to verify the routines.

Model Development

Two models were developed for the short period mode of motion. First, the short period approximation was used to produce a differential equation representation of the short period dynamics. The differential equations were then transferred into state space format for use with PC Matlab. The second model started with the general form of a short period pitch rate to elevator deflection ($q/\delta e$) transfer function. The primary difference was that the transfer function model estimated second order system parameters (damping ratio, natural frequency, etc.) directly, whereas the differential equation model estimated stability parameters in the aircraft equations of motion.

Short Period Approximation. The longitudinal small perturbation equations of motion can be expressed in many

forms. For simplicity of expression, Roskam (14) develops the equations in the stability axis system (14:391). The equations are developed assuming perturbations around a steady state condition for which there is no initial side velocity or bank angle, and there are no initial angular velocities (14:45). For convenience the elements of the equations are also defined in terms of dimensional stability parameters (14:413). These equations are expressed in differential equation form (14:414) as

$$\dot{u} = - g\theta\cos\theta_1 + X_u u + X_{Tu} u + X_\alpha \alpha + X_{\delta e}\delta e \qquad (13)$$

$$\dot{w} - U_1 q = - g\theta\sin\theta_1 + Z_u u + Z_\alpha \alpha + Z_{\dot{\alpha}}\dot{\alpha} + Z_q q + Z_{\delta e}\delta e \qquad (14)$$

and

$$\dot{q} = M_u u + M_{Tu} u + M_\alpha \alpha + M_{T\alpha}\alpha + M_{\dot{\alpha}}\dot{\alpha} + M_q q + M_{\delta e}\delta e \qquad (15)$$

For the short period approximation, several assumptions are made to simplify the model. First of all, the short period mode is assumed to occur at constant forward speed (u), so the u portions of the equations can be deleted. Substituting $\dot{\alpha} = \dot{w}/U_1$, the resulting equations are

$$(U_1 - Z_{\dot{\alpha}})\dot{\alpha} = - g\sin\theta_1 + Z_\alpha \alpha + (U_1 + Z_q) q + Z_{\delta e}\delta e \qquad (16)$$

and

$$\dot{q} = M_{\alpha}\alpha + M_{T\alpha}\alpha + M_{\dot{\alpha}}\dot{\alpha} + M_{q}q + M_{\delta e}\delta e \qquad (17)$$

If the steady state condition is defined as trimmed level flight, then $\theta_1 = 0$. In addition, it is further assumed that $Z_{\dot{\alpha}}$ and Z_q are much smaller than U_1. Finally, it is assumed that the $M_{T\alpha}$ derivative is included in M_{α}. In other words, the combined effect of M_{α} and $M_{T\alpha}$ is included in the single derivative M_{α}. Incorporating these assumptions yields the following set of equations:

$$U_1\dot{\alpha} = Z_{\alpha}\alpha + U_1 q + Z_{\delta e}\delta e \qquad (18)$$

and

$$\dot{q} = M_{\alpha}\alpha + M_{\dot{\alpha}}\dot{\alpha} + M_{q}q + M_{\delta e}\delta e \qquad (19)$$

One additional assumption is made at this point. While an analytical distinction can be made between the $\dot{\alpha}$ and q derivatives, flight test produces poor results when attempting to separate these derivatives in a particular maneuver (11:17). Therefore, the $\dot{\alpha}$ and q derivatives are combined into a single q derivative. In other words,

$$M_q q + M_{\dot{\alpha}}\dot{\alpha} \rightarrow M_q q \qquad (20)$$

With this final assumption, the equations can now be rewritten solving for $\dot{\alpha}$ and $\dot{q}$. This results in the following set of equations:

$$\dot{\alpha} = (Z_\alpha/U_1)\alpha + q + (Z_{\delta e}/U_1)\delta e \tag{21}$$

and

$$\dot{q} = M_\alpha \alpha + M_q q + M_{\delta e}\delta e \tag{22}$$

Or, in state space representation

$$\begin{Bmatrix} \dot{\alpha} \\ \dot{q} \end{Bmatrix} = \begin{bmatrix} Z_\alpha/U_1 & 1 \\ M_\alpha & M_q \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} Z_{\delta e}/U_1 \\ M_{\delta e} \end{Bmatrix} \delta e \tag{23}$$

In this case the outputs are the states themselves (angle of attack and pitch rate), so

$$y = \begin{Bmatrix} \alpha \\ q \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \delta e \tag{24}$$

For the purposes of checking out the simulation, dimensional stability derivatives were used for the A-4D aircraft. A middle of the envelope flight condition was chosen at 15,000 feet pressure altitude and 0.60 Mach number. Table 1 below shows the A-4D data used (9:702).

Table 1. A-4D Simulation Data

h = 15,000 ft	Mach = 0.60
U_1 = 634 ft/sec	Z_α = -518.6 (ft/sec^2)/rad
M_α = -12.68 (1/sec^2)	$Z_{\delta e}$ = -56.92 (ft/sec^2)/rad
M_q = -1.07 (1/sec)	$M_{\delta e}$ = -19.48 (1/sec^2)

Rewriting equation (23) with the specific value of U_1 yields

$$\begin{Bmatrix} \dot{\alpha} \\ \dot{q} \end{Bmatrix} = \begin{bmatrix} (.00158\ Z_\alpha) & 1 \\ M_\alpha & M_q \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} (.00158\ Z_{\delta e}) \\ M_{\delta e} \end{Bmatrix} \delta e \qquad (25)$$

Now for purposes of parameterizing tne model for the MMLE subroutine, the stability derivatives in equation (25) are represented as follows:

$$p(1) = Z_\alpha$$

$$p(2) = M_\alpha$$

$$p(3) = M_q$$

$$p(4) = Z_{\delta e}$$

$$p(5) = M_{\delta e}$$

So, the final form of the model for the MMLE subroutine is

$$\begin{Bmatrix} \dot{\alpha} \\ \dot{q} \end{Bmatrix} = \begin{bmatrix} .00158*p(1) & 1 \\ p(2) & p(3) \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} .00158*p(4) \\ p(5) \end{Bmatrix} \delta e \qquad (26)$$

$$y = \begin{Bmatrix} \alpha \\ q \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \delta e \qquad (27)$$

Equations (26) and (27) represent the final form of the short period approximation model which was used for the MMLE software checks using simulated data.

Transfer Function Model. In addition to the short period approximation model, a second method was used to provide a cross check of the results from the short period approximation. This second method estimated second order system parameters directly, and the model development started with the short period approximation pitch rate to elevator deflection ($q/\delta e$) transfer function:

$$\frac{q(s)}{\delta e(s)} = \frac{K_\theta (s+1/T_{\theta 2})}{s^2+2\zeta_{sp}\omega_{sp}s+\omega_{sp}^2} \qquad (28)$$

A set of state variables is required so that this transfer function can be expressed in the form $\dot{x} = Ax + Bu$. Since the transfer function could be rewritten in differential equation form, the resulting differential equation was examined to help define state variables. Equation (28) can be rewritten as

$$\left(s^2 + 2\zeta_{sp}\omega_{sp}s + \omega_{sp}^2\right)q(s) = K_\theta(s + 1/T_{\theta 2})\delta e(s) \qquad (29)$$

Taking the inverse Laplace transform yields

$$\ddot{q} + 2\zeta_{sp}\omega_{sp}\dot{q} + \omega_{sp}^2 q = K_\theta\dot{\delta}e + K_\theta(1/T_{\theta 2})\delta e \qquad (30)$$

Equation (30) is the differential equation representation of the short period q/δe transfer function. Normally q and $\dot{q}$ could be used for state variables since knowledge of q(0) and $\dot{q}$(0) along with δe(t) for t $\geq$ 0 completely determines the response of the system. However, a problem arises due to the presence of the derivative term of the forcing function δe(t) in equation (30). Ogata (12) presents a method of choosing state variables whereby the control derivative term is eliminated from the state equation (12:675-678).

If a set of state variables

$$x_1 = q$$

and

$$x_2 = \dot{q} - K_\theta\delta e$$

is chosen, then the state and output equations for the differential equation in equation (30) are

$$\dot{x}_1 = \dot{q} = x_2 + K_\theta \delta e \tag{31}$$

$$\dot{x}_2 = \ddot{q} - K_\theta \dot{\delta} e = -\omega_{sp}^2 q - 2\zeta_{sp}\omega_{sp}\dot{q} + K_\theta(1/T_{\theta 2})\delta e$$

$$= -\omega_{sp}^2 x_1 - 2\zeta_{sp}\omega_{sp} x_2 + K_\theta(1/T_{\theta 2} - 2\zeta_{sp}\omega_{sp})\delta e \tag{32}$$

and

$$y = x_1 \tag{33}$$

Rewritten in matrix form, equations (31) through (33) become

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_{sp}^2 & -2\zeta_{sp}\omega_{sp} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} + \begin{Bmatrix} K_\theta \\ K_\theta(1/T_{\theta 2} - 2\zeta_{sp}\omega_{sp}) \end{Bmatrix} \delta e \tag{34}$$

and

$$y = [1 \;\; 0] \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} \tag{35}$$

Now to parameterize the model for MMLE, the terms in equation (34) are redefined as follows:

$$p(1) = K_\theta$$

$$p(2) = 1/T_{\theta 2}$$

$$p(3) = 2\zeta_{sp}\omega_{sp}$$

$$p(4) = \omega_{sp}^2$$

Therefore, the final model used for the pitch rate transfer function model with MMLE is

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix} = \begin{bmatrix} 0 & 1 \\ -p(4) & -p(3) \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} + \begin{Bmatrix} p(1) \\ p(1) * (p(2) - p(3)) \end{Bmatrix} \delta e \tag{36}$$

and

$$y = q = [1 \ 0] \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} \tag{37}$$

For the purposes of checking out the simulation, the transfer function parameters were calculated based on the dimensional stability derivatives for the A-4D aircraft. These derivatives were shown in Table 1 above. Starting with the Laplace transforms of equations (21) and (22),

$$s\alpha(s) = (Z_\alpha / U_1)\alpha(s) + q(s) + (Z_{\delta e} / U_1)\delta e(s) \tag{38}$$

and

$$sq(s) = M_\alpha \alpha(s) + M_q q(s) + M_{\delta e} \delta e(s) \tag{39}$$

or in matrix form,

$$\begin{bmatrix} (s - Z_\alpha / U_1) & -1 \\ -M_\alpha & (s - M_q) \end{bmatrix} \begin{Bmatrix} \alpha(s) \\ q(s) \end{Bmatrix} = \begin{Bmatrix} Z_{\delta e} / U_1 \\ M_{\delta e} \end{Bmatrix} \delta e(s) \tag{40}$$

Using Cramer's rule on equation (40) yields

$$\frac{q(s)}{\delta e(s)} = \frac{[s-(Z_\alpha/U_1)]M_{\delta e} + M_\alpha(Z_{\delta e}/U_1)}{[s-(Z_\alpha/U_1)](s-M_q) - M_\alpha} \qquad (41)$$

Equation (41) can be rewritten as

$$\frac{q(s)}{\delta e(s)} = \frac{M_{\delta e}s + [M_\alpha(Z_{\delta e}/U_1) - M_{\delta e}(Z_\alpha/U_1)]}{s^2 - [(Z_\alpha/U_1)+M_q]s + [M_q(Z_\alpha/U_1) - M_\alpha]} \qquad (42)$$

Substituting the A-4D data from Table 1,

$$\frac{q(s)}{\delta e(s)} = \frac{-19.48\,(s + 0.75954)}{s^2 + 1.88798s + 13.55524} \qquad (43)$$

Equation (43) is now in the standard form of equation (28). The transfer function parameters of interest (K_θ, $1/T_{\theta 2}$, ζ_{sp}, and ω_{sp}) can be calculated directly from equation (43). The parameters calculated using the A-4D data are shown in Table 2 below. These numbers will be used to compare with the MMLE estimates obtained using the transfer function model.

Table 2. A-4D Calculated Transfer Function Data

h = 15,000 ft	Mach = 0.60
K_θ = -19.48	$1/T_{\theta 2}$ = 0.7595
ζ_{sp} = 0.2564	ω_{sp} = 3.6817

Input and Output Generation

The type of input used for the parameter estimation maneuver is critical to the success of the process. Past studies have shown that inputs with sharp corners produced the best results. Sets of derivatives have been obtained using simple doublet inputs, which produce results as good as those from more complex inputs (15:1).

For the longitudinal axis, the maneuver of choice at the AFFTC is a rapid elevator doublet. The doublet is followed by approximately 5 to 10 seconds of free response. The control derivatives are identified during the input time segment, and stability and damping derivatives are identified during the free response.

As described by Nagy (11:31), derivatives in general are functions of angle of attack and Mach number, and these are the key parameters which should be held as constant as possible throughout the maneuver. Although angle of attack will obviously vary with an elevator doublet, the doublet should be short enough in duration that no large pitch rate or angle of attack perturbations occur. Then at least the derivatives will be valid for the average angle of attack of the maneuver. As for Mach number, the importance of maintaining trim conditions is a primary reason for using a doublet input instead of a pulse (11:31).

The doublet input should have a magnitude such that the response is as large as possible without violating the assumptions of the linear model. Larger inputs result in higher signal-to-noise ratios in the data, and hence yield better results (15:6). As a rough guideline, pitch rate excursions should be around 10 to 20 degrees per second peak to peak. Angle of attack should vary within about four degrees peak to peak (11:31).

A triangular doublet was designed using PC Matlab, with variable amplitude and frequency. For this simulation, the frequency was chosen to give a complete doublet input in one second. The amplitude was chosen as ±0.07 radians of elevator deflection (approximately ±4 degrees). The input was designed with 20 samples per second to avoid problems in determining the effects of an input at this frequency. The total maneuver duration was 10 seconds.

The "true" values of each parameter (shown in Table 1) for the A-4D were used to generate the output of the system With the Table 1 values, the resulting state space system is

$$\begin{Bmatrix} \dot{\alpha} \\ \dot{q} \end{Bmatrix} = \begin{bmatrix} -.81939 & 1 \\ -12.68 & -1.07 \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} -.08993 \\ -19.48 \end{Bmatrix} \delta e \qquad (44)$$

$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \delta e \qquad (45)$$

The noise free output response data for pitch rate and angle of attack are generated using the DLSIM function in PC Matlab (6:CR-32). The DLSIM function simulates the outputs of a discrete time linear system. The state space system in equations (44) and (45) was discretized using the C2D continuous to discrete conversion function of PC Matlab (6:CR-11). The C2D function requires the system A and B matrices, and the time increment as inputs. It returns the Φ and Γ matrices in the corresponding discrete time system:

$$x[n+1] = \Phi x[n] + \Gamma u[n] \tag{46}$$

assuming the control input is piecewise constant over the time increment. The DLSIM function takes the Φ and Γ matrices, along with the output C and D matrices and the defined input, and returns the simulated time history data for the specified system and input.

The state space system matrices of equations (44) and (45) were processed through the continuous to discrete conversion function, and subsequently through the discrete linear systems simulation. Figure 1 shows the PC Matlab code required to define the input, input the system matrices, and produce the time history output. Figure 2 shows the simulated time histories of elevator deflection, angle of attack, and pitch rate, respectively. The noise free simulated data was then used to accomplish the

```
'-------------------------------------------------------';
disp(ans)
disp('            INPUT/OUTPUT SIMULATION MACRO')
disp('       A-4D LONGITUDINAL STABILITY PARAMETERS')
disp('  TRUE DATA = [-518.6 -12.68 -1.07 -56.92 -19.48]')
disp(ans)
%
%-----------------------------------------SIMULATE INPUT
%
ndp=201;        %   INPUT NUMBER OF DATA POINTS
dt=.05;         %   INPUT TIME INCREMENT
amp=.07;        %   INPUT AMPLITUDE OF DOUBLET IN RADIANS
per=1;          %   INPUT PERIOD OF DOUBLET IN SECONDS
%
t=[0:ndp-1]*dt;                 % THESE LINES
uydata=zeros(ndp,3);            % GENERATE THE
inc=(4*amp*dt)/(per);           % ELEVATOR DOUBLET
u1=(0:-1*inc:-1*amp);           % FROM THE USER'S
u2=((-1*amp)+inc:inc:amp);      % SPECIFIED PERIOD
u3=(amp-inc:-1*inc:0);          % AND AMPLITUDE
uydata(:,1)=[u1 u2 u3 zeros(1,ndp-(per/dt)-1)]';
%
%----------------------------------------SIMULATE OUTPUT
%
%          INPUT SYSTEM MATRICES FOR SIMULATION
%
a=[-.81939     1
   -12.68    -1.07];
%
b=[-.08993
   -19.48 ];
%
c=[   1        0
      0        1  ];
%
d=[   0
      0    ];
%
%          UYDATA(:,2) = ANGLE OF ATTACK (RAD)
%          UYDATA(:,3) = PITCH RATE (RAD/SEC)
%
[phi,gam]=c2d(a,b,dt); %                          DISCRETIZE
%
uydata(:,2:3)=dlsim(phi,gam,c,d,uydata(:,1));%NOISE-FREE
save uydata %                                      RESPONSE
%-----------------------------------------END SIMULATE.M
```

Figure 1. PC Matlab Data Simulation Code

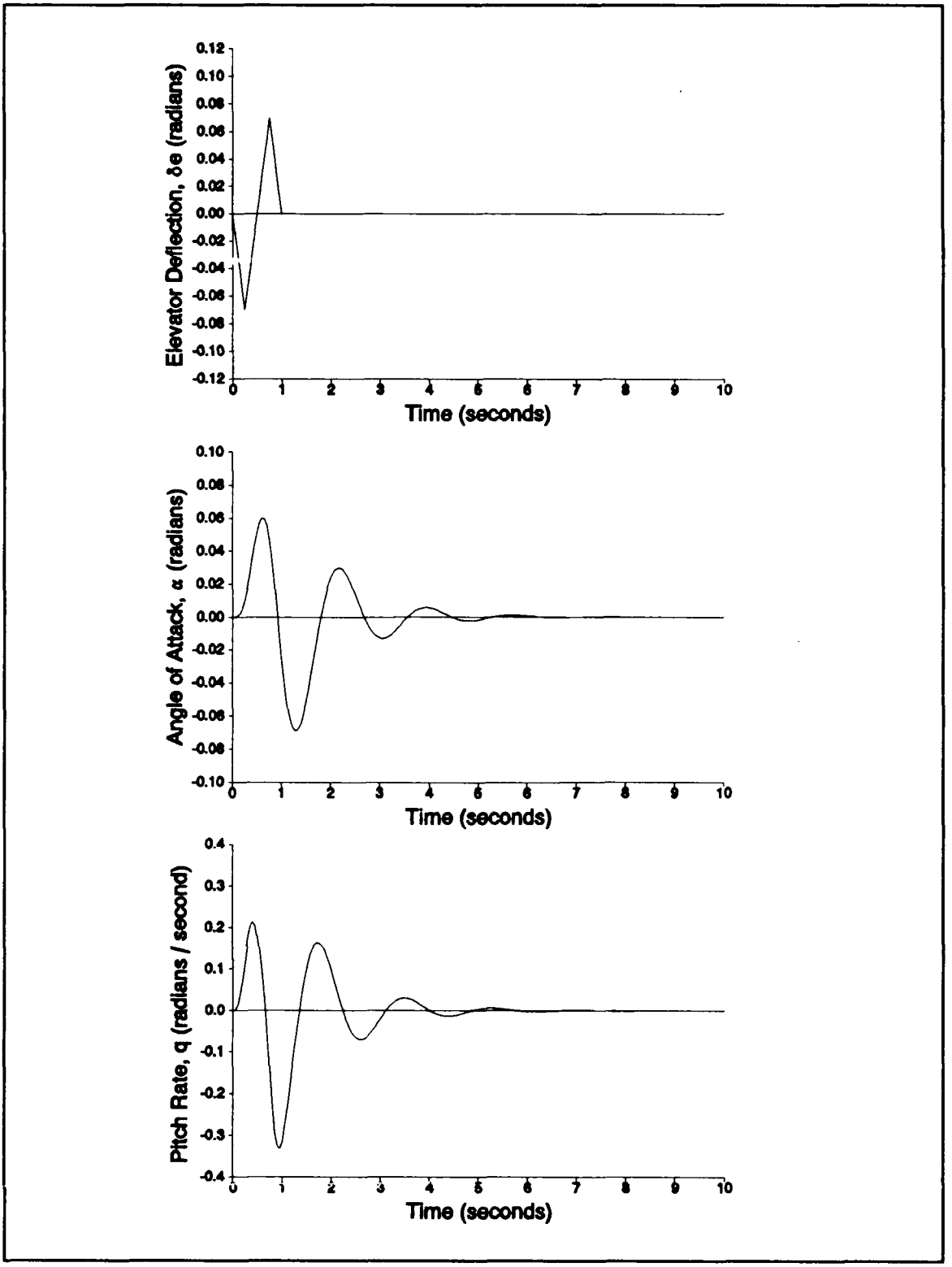

Figure 2. Simulated Data Time Histories

parameter estimation task through both the short period approximation and transfer function models.

PC Matlab Macro File Creation

Using the PC Matlab MMLE program to accomplish parameter identification involves basically three steps. First, the user must create a file to convert the parameter vector into a state space system model. Secondly, the necessary inputs to the MMLE subroutine must be defined. Finally, the MMLE macro file is executed.

The first step is the creation of the parameter-to-state-space (P2SS) macro function. This function primarily converts the parameter vector (p) into the user-defined state space description of the system. In addition to the state space system matrices (A, B, C, and D), the function must return the following:

1. **q** = square root of the state noise covariance matrix.
2. **rowinq** = a vector (the same size as p) showing where parameters to be estimated occur in the q matrix.
3. **x0** = a row vector containing initial conditions of the states.
4. **dt** = the time increment for discretizing the continuous model.

5. **phi** = Φ, the state transition matrix.
6. **gam** = Γ, the discrete control distribution matrix.

The Φ and Γ matrices can be obtained from the continuous to discrete transformation (C2D) already discussed.

Once the P2SS file is created, several other necessary inputs to MMLE must be defined. The easiest way is to create another macro file for defining the required inputs to MMLE. The following are the necessary inputs (10:SR-6,SR-7):

1. **uydata** = a matrix consisting of columns corresponding to the time histories of first the inputs, then the outputs.
2. **p2snam** = a string containing the file name of the P2SS function.
3. **p0** = a row vector containing initial values of the parameters to be identified.
4. **rms0** = a vector (the same size as p0) containing standard deviations of a-priori estimates if the a-priori weighting option is used. The **rms0** value defaults to zero if it is undefined.
5. **pref** = a vector (the same size as p) containing reference, or true, values for each of the parameters.

6. **pidq** = a row vector giving the locations of parameters in the vector p which should be identified in the quadratic phase.
7. **pidm** = a row vector giving the locations of parameters in the vector p which should be identified in the Marquardt phase.
8. **pidf** = a row vector giving the locations of parameters in the vector p which should be identified in the constrained Newton and "gg determination" phases.
9. **pert** = the perturbation to be used for numerical calculation of gradients.
10. **gg0** = the initial guess of the innovations covariance matrix.
11. **diagrr** = a flag which, if defined, zeroes off-diagonal terms in the estimated innovation covariance matrix.
12. **linesearch** = 1 enables the linesearch along the Newton direction. Linesearch defaults to zero if it is not defined.
13. **opt** = a vector which controls the number of iterations and criteria for convergence. The **opt** default is [0 5 5 5 .02 .05 .001 1]. The elements of the vector (in sequence) are as follows:

a. **qstep** = the number of quadratic steps.

b. **mstep** = maximum number of Marquardt steps.

c. **nstep** = maximum number of constrained Newton steps.

d. **gstep** = maximum number of "gg determination" steps.

e. **marqf** = the value of parameter **marq** which terminates the Marquardt phase.

f. **fconv** = the convergence criterion for early exit from the constrained Newton phase.

g. **gconv** = final convergence criterion.

h. **rhz** = frequency of a low pass filter used to compute filtered Cramér-Rao bounds.

The four minimization phases discussed above require further description (10:SR-7,SR-8). The quadratic phase is a pure Newton phase which can find the minimum of a quadratic cost function in one iteration. This phase is skipped in the default **opt** vector. The Marquardt phase varies the step size along the gradient to ensure that the problem is well conditioned for the constrained Newton phase. The constrained Newton phase is the final optimization phase which uses the Newton algorithm. The "gg determination" phase is used for estimating the gg matrix.

Once the necessary inputs are defined, completing the parameter identification process is simply a matter of executing the MMLE macro. Having addressed the requirements for running MMLE in PC Matlab, the specific macro files for accomplishing parameter identification with the two models can be discussed.

Short Period Approximation Macro. Recall from equations (26) and (27) the final form of the short period approximation model parameterized for use with MMLE. The equations are repeated here for convenience:

$$\begin{Bmatrix} \dot{\alpha} \\ \dot{q} \end{Bmatrix} = \begin{bmatrix} .00158*p(1) & 1 \\ p(2) & p(3) \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} .00158*p(4) \\ p(5) \end{Bmatrix} \delta e \qquad (47)$$

and

$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \alpha \\ q \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \delta e \qquad (48)$$

The macro file SPP2SS.M was created to perform the parameter-to-state-space conversion for the short period approximation model. The PC Matlab code in the macro file is shown in Figure 3. The matrix definitions are straight forward. Since the MMLE run is to be performed against simulated data, the state noise matrix Q is set to a small non-zero diagonal matrix (basically eliminating its effect

```
function[a,phi,gam,c,d,q,x0,dt,rowinq,b]=spp2ss(p);
%-------------------------------------------------------
% P2SS FUNCTION FOR SHORTPD.M ---- EXTENSIVELY COMMENTED
% STATE-SPACE REALIZATION OF SHORT PERIOD APPROXIMATION
% WITH A-4D STABILITY PARAMETERS (H = 15,000 FT, M = 0.6)
%-------------------------------------------------------
%
% p(1) = Z_ALPHA
% p(2) = M_ALPHA
% p(3) = M_Q
% p(4) = Z_DE
% p(5) = M_DE
%
%---------------------------------------- SYSTEM MATRICES
%
a=[.00158*p(1)  1
      p(2)     p(3)];
%
b=[.00158*p(4)
      p(5)    ];
%
c=[  1  0
     0  1  ];
%
d=[  0
     0  ];
%
%---------------------------------- STATE NOISE COVARIANCE
%
q=eye(2)*1e-6;%              Q IS THE SAME SIZE AS A WITH
              %                   Q*Q' POSITIVE DEFINITE!
%
%-------------------- ROWS IN Q IN WHICH PARAMETERS OCCUR
%
rowinq=0*p;
%
%---------------------------------- INITIAL STATE VECTOR
%
x0=zeros(1,2);
%
%------------------------------ DEFINE DT AND DISCRETIZE
%
dt=.05;
[phi,gam]=c2d(a,b,dt);
%
%------------------------------------------ END SPP2SS.M
```

Figure 3. Short Period P2SS Function Code

on the problem). There are, therefore, no parameters to be identified in the Q matrix, so the **rowinq** variable is a vector of zeroes. The simulated data started with zero initial conditions, so the **x0** vector is also set to zero. Finally, the macro file defines the time increment from the data simulation and makes the call to the continuous to discrete (C2D) function to calculate the Φ and Γ matrices.

Once the SPP2SS.M file is created, the required inputs discussed above have to be defined. This purpose is served by the SHORTPD.M file, shown in Figure 4. This macro is set up to input the true parameters for the model (previously shown in Table 1), and load the input and output data file created by the simulation macro defined above. The **pidq**, **pidm**, and **pidf** parameters are defined such that all five parameters for the identification will be identified in the constrained Newton phase. Even though the quadratic and Marquardt phases are not used, the corresponding parameters (**pidq** and **pidm**) must be defined. Therefore, they are set to a nominal value of one, while **pidf** is set to reflect identification of parameters one through five.

The initial parameter estimate is set up with a 30 percent error from the true parameters. The default **opt** vector is redefined to reflect parameter identification in the constrained Newton phase only, and has a slightly more stringent requirement on final convergence. The **gg0** matrix

```
format compact,clc
clear;
diary shortpd.log
'-------------------------------------------------------';
disp('          SHORT PERIOD APPROXIMATION MODEL')
disp('       A-4D LONGITUDINAL STABILITY PARAMETERS')
disp('  TRUE DATA = [-518.6 -12.68 -1.07 -56.92 -19.48]')
%
%--------------------------- GET TRUE STABILITY PARAMETERS
%
% THE RESULTS MATRIX PRODUCED BY MMLE.M USES PREF IF
% AVAILABLE
%
pref=[-518.6 -12.68 -1.07 -56.92 -19.48];
%
%------------------------ SUFFICIENT INPUTS TO MMLE FOLLOW
%
load uydata; % UYDATA IS OUTPUT FROM THE SIMULATE.M MACRO,
%               AND MUST BE AVAILABLE AS A DATA FILE.
%
pidq=[1]; %  IDENTIFY WHICH PARAMETERS ARE TO BE ESTIMATED
pidm=[1]; %   IN THE QUADRATIC (PIDQ), MARQUARDT (PIDM),
pidf=[1:5];% AND CONSTRAINED NEWTON (PIDF) PHASES
%
p0=0.7*pref;%OFFSET INITIAL GUESS 30% FROM TRUE PARAMETERS
%
opt=[0 0 5 0 .02 .01 .001 1]; %IF YOU SIMULATED NOISE-FREE
%                                DATA,  MAKE SURE OPT(4)=0!
%
gg0=1e-4*eye(2); % INITIAL GUESS OF INNOVATIONS COVARIANCE
%
p2snam='spp2ss'; %              FILENAME OF P2SS FUNCTION
%
pert=1e-4; %     SET PERTURBATION FOR GRADIENT CALCULATION
%
linesearch=1;
mmle %                     SUBROUTINE CALL TO MMLE PROGRAM
%
Z_ALPHA=pfin(1) %     PRINT OUT FINAL STABILITY PARAMETERS
M_ALPHA=pfin(2)
M_Q=pfin(3)
Z_DE=pfin(4)
M_DE=pfin(5)
%
diary off
%-----------------------------------------END SHORTPD.M
```

Figure 4. Short Period MMLE Macro Code

is defined, as with Q, as a small non-zero valued diagonal matrix. Finally, the linesearch flag is set and the subroutine call to MMLE is made. The remaining lines in the macro file are for final printout of the parameter identification results.

Transfer Function Macro. The final form of the transfer function model for use with the MMLE program was expressed in equations (36) and (37), and is repeated here for convenience:

$$\begin{Bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{Bmatrix} = \begin{bmatrix} 0 & 1 \\ -p(4) & -p(3) \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} + \begin{Bmatrix} p(1) \\ p(1) * \{p(2) - p(3)\} \end{Bmatrix} \delta e \qquad (49)$$

and

$$y = [1 \ 0] \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} \qquad (50)$$

For the transfer function model, the QDEP2SS.M file sets up the parameter-to-state-space conversion. As with the short period approximation model, the file defines the state space matrices in terms of the parameter vector, and the remaining variable definition is identical to that in the SPP2SS.M file. The QDEP2SS.M macro file is shown in Figure 5.

As with the short period approximation model, a macro file performs all of the required input definition for MMLE.

```
function[a,phi,gam,c,d,q,x0,dt,rowinq,b]=qdep2ss(p);
%-------------------------------------------------------------
% P2SS FUNCTION FOR QDELOES.M ---- EXTENSIVELY COMMENTED
% STATE-SPACE PITCH RATE TO ELEVATOR X-FER FCN
%-------------------------------------------------------------
%                   Q      A_THETA * (S + 1/T_THETA2)
%                  --- = ----------------------------
%                  dE       S^2 + 2*ZETA*WN*S + WN^2
%-------------------------------------------------------------
%
% p(1) = A_THETA
% p(2) = 1/T_THETA2
% p(3) = 2*ZETA*WN
% p(4) = WN^2
%
%---------------------------------------- SYSTEM MATRICES
%
a=[     0         1
    -1*p(4)  -1*p(3)];
%
b=[         p(1)
    p(1)*(p(2)-p(3))];
%
c=[     1         0   ];
%
d=0;
%
%---------------------------------- STATE NOISE COVARIANCE
%
q=eye(2)*1e-6;%                Q IS THE SAME SIZE AS A WITH
%                               Q*Q' POSITIVE DEFINITE!
%
%---------------------- ROWS IN Q IN WHICH PARAMETERS OCCUR
%
rowinq=0*p;
%
%----------------------------------- INITIAL STATE VECTOR
%
x0=zeros(1,2);
%
%------------------------------- DEFINE DT AND DISCRETIZE
%
dt=.05;
[phi,gam]=c2d(a,b,dt);
%
%-----------------------------------------ENDQDEP2SS.M
```

Figure 5. Transfer Function P2SS Function Code

The QDELOES.M file (shown in Figure 6) defines all the necessary variables, with the same assumptions and definition used in the SHORTPD.M macro file used for the short period approximation model. The primary difference between the two is that the QDELOES.M file calculates the equivalent second-order system parameters (K_θ, $1/T_{\theta 2}$, ζ_{sp}, and ω_{sp}), and requires only pitch rate as an output parameter.

Simulation Results

The simulated input and output data described above were processed through both the short period approximation and the transfer function models. Since the data were simulated, and produced responses with no noise content, one would expect the MMLE program to be able to match the time history responses exactly. That is, in fact, what happened. For both models, the predicted time history data from the MMLE program were identical to the "actual" simulated time histories.

Since the data were simulated using the data for the A-4D in the short period approximation model, one would also expect the parameters identified to match exactly. That was also the case, within very tight tolerances. The A-4D parameters were listed in Table 1. They are compared with the results from the short period approximation run in Table 3 below.

```
format compact,clc
clear;
diary qdeloes.log
'-------------------------------------------------------';
disp('    PITCH RATE TO ELEVATOR EQUIVALENT SYSTEM MATCH')
disp('          A-4D LONGITUDINAL SYSTEM PARAMETERS')
disp('     TRUE DATA = [-19.48 .75954 1.88798 13.55524]')
%
%-----------------------------------------SIMULATEDATA
%
% THE RESULTS MATRIX PRODUCED BY MMLE.M USES PREF IF
% AVAILABLE
%
pref=[-19.48 .75954 1.88798 13.55524]; %   GET TRUE SYSTEM
%                                            PARAMETERS
%
%------------------------ SUFFICIENT INPUTS TO MMLE FOLLOW
%
uydata=[uydata(:1) uydata(:,3)];% USES INPUT/OUTPUT VECTOR
%                                  CREATED BY SHORT PERIOD
%                                   SIMULATION, WITH ONLY
%                                  PITCH RATE AS AN OUTPUT
%
pidq=[1]; %  IDENTIFY WHICH PARAMETERS ARE TO BE ESTIMATED
pidm=[1:4];% IN THE QUADRATIC (PIDQ), MARQUARDT (PIDM),
pidf=[1:4];% AND CONSTRAINED NEWTON (PIDF) PHASES
%
p0=0.7*pref;%OFFSET INITIAL GUESS 30% FROM TRUE PARAMETERS
%
opt=[0 0 5 0 .02 .01 .001 1];% IF YOU SIMULATED NOISE-FREE
%                               DATA, MAKE SURE OPT(4)=0!
%
gg0=1e-4; %       INITIAL GUESS OF INNOVATIONS COVARIANCE
%
p2snam='qdep2ss'; %             FILENAME OF P2SS FUNCTION
%
pert=1e-4; %    SET PERTURBATION FOR GRADIENT CALCULATION
linesearch=1;
mmle %                     SUBROUTINE CALL TO MMLE PROGRAM
%
a_theta=pfin(1) %       PRINT OUT FINAL SYSTEM PARAMETERS
invt_theta2=pfin(2)
zetasp=pfin(3)/(2*sqrt(pfin(4)))
wsp=sqrt(pfin(4))
diary off
%-----------------------------------------ENDQDELOES.M
```

Figure 6. Transfer Function MMLE Macro Code

Table 3. Short Period MMLE Results

PARAMETER	"TRUE" DATA	MMLE DATA
Z_α	-518.60	-518.6013
M_α	-12.68	-12.6800
M_q	-1.07	-1.0700
$Z_{\delta e}$	-56.92	-56.9177
$M_{\delta e}$	-19.48	-19.4800

The MMLE run using the transfer function model also matched the time histories exactly. The parameters identified using this model were compared with those calculated using Cramer's rule on the short period approximation system matrices. Those calculations were listed in Table 2. Table 4 below summarizes the MMLE results from the transfer function model.

Table 4. Transfer Function MMLE Results

PARAMETER	"TRUE" DATA	MMLE DATA
K_θ	-19.4800	-19.4800
$1/T_{\theta 2}$	0.7595	0.7609
ζ_{sp}	0.2564	0.2566
ω_{sp}	3.6817	3.6819

Having achieved such good results with the simulated data, the MMLE routines were considered to be checked out

and ready to use with flight test data. However, several issues surfaced prior to flight test which required modification of the longitudinal model. The next section discusses the reasons behind the changes, and the subsequent model which was developed for use in flight test.

IV. Preparations for Flight Test

Model Changes

Several issues arose prior to flight test which brought about changes in the model. The primary reasons were violated assumptions and convenience for flight test parameter measurements. The specifics behind the changes are discussed in this subsection.

As previously mentioned, the equations of motion used for the short period approximation model were derived with several assumptions. First of all, they were developed in terms of dimensional stability parameters. Non-dimensional stability derivatives are the preferred form, because they facilitate comparisons among different types of aircraft, as well as different flight conditions. A form of the model using these non-dimensional stability derivatives directly was considered preferable to that developed for the simulation.

The previous model was also developed using the stability axis system. The parameters measured in flight test are normally measured with respect to the body axis of the aircraft. In addition, the data available for moments and products of inertia are normally referenced to the body axis. While the transformation from stability to body axis is a simple rotation through the aircraft angle of attack,

it was considered preferable to express the model in terms of the aircraft body axis to simplify the calculations required.

The non-linear equations of motion were linearized about a steady state condition of trimmed, wings level flight. One of the obvious assumptions associated with this condition is that there is no initial bank angle. For performing the required elevator doublets from trimmed, wings level flight, this assumption would be valid. However, as will be discussed in the flight test section, one of the methods for performing these maneuvers includes steady state conditions of trimmed, elevated load factor turns. These conditions are used to obtain flight test data at incremental angles of attack above trim, while maintaining a particular flight condition. In this case, the assumption of no initial bank angle is violated, and the relationships among parameters in the equations of motion are more complex.

From the above issues, it can be seen that, whether out of convenience or necessity, changes are required in the model prior to flight test. Since the MMLE3 program is already accepted as the flight test standard for parameter identification, it is a good starting point for the longitudinal aircraft model.

The New Model

The longitudinal equations of motion developed in the body axis for use in MMLE3 are (11:206;7:44)

$$\dot{\alpha} = q - p\tan\beta\cos\alpha + \frac{g\cos\theta\cos\phi}{V\cos\alpha\cos\beta}$$

$$- \frac{\bar{q}S}{mV\cos\alpha\cos\beta}(C_{No}+C_{N\alpha}\alpha+C_{N\delta e}\delta e) \qquad (51)$$

$$\dot{q} = \frac{1}{I_{yy}}[(I_{zz}-I_{xx})pr + I_{xy}(\dot{p}+qr) + I_{xz}(r^2-p^2)$$

$$+ I_{yz}(\dot{r}-pq) + \bar{q}Sc(C_{Mo}+C_{M\alpha}\alpha+C_{M\delta e}\delta e+\frac{c}{2V}(C_{Mq}q))] \qquad (52)$$

and

$$\dot{\theta} = q\cos\phi - r\sin\phi + \dot{\theta}_0 \qquad (53)$$

Since the state equations contain terms which are neither functions of the state variables nor the control input, the control vector is augmented with an additional input to incorporate these terms. It should also be pointed out that the terms such as C_{No}, C_{Mo}, and $\dot{\theta}_o$ in these equations are also used to correct for instrumentation measurement biases. Some problems were encountered in the initial development of

the model, when considering the variables as perturbations from trim. If these terms are not included in the model, any instrumentation biases translate directly into initial rates in the model. This problem is overcome by maintaining the variables as total measurements, with the appropriate bias terms included in the model.

Equations (51) through (53) represent a two degree of freedom model, where the forward velocity equation is omitted. The derivatives in that equation are normally obtained from other sources. These equations are non-linear, and several assumptions must be made to reduce them to a form usable by the MMLE routine. It is first assumed that there is negligible coupling with the lateral-directional axes. In other words, for a pure elevator doublet, p = r = 0. It is also assumed that the x-z plane is a plane of symmetry, such that $I_{xy} = I_{yz} = 0$. Sideslip angle is assumed small enough to use the small angle approximation that cos β = 1. Finally, in order to preserve the linearity of the model, the trigonometric functions in equations (51) through (53) are assumed to be constant, and are evaluated at the initial conditions. With these assumptions, the equations become

$$\dot{\alpha} = q + \frac{g\cos\theta_1\cos\phi_1}{V\cos\alpha_1} - \frac{\bar{q}S}{mV\cos\alpha_1}\left(C_{No}+C_{N\alpha}\alpha+C_{N\delta e}\delta e\right) \qquad (54)$$

$$\dot{q} = \frac{\bar{q}Sc}{I_{yy}} \left[C_{Mo} + C_{M\alpha}\alpha + C_{M\delta e}\delta e + \frac{c}{2V}(C_{Mq}q) \right] \quad (55)$$

and

$$\dot{\theta} = q\cos\phi_1 + \dot{\theta}_0 \quad (56)$$

The corresponding measurement equations (7:45) are

$$\alpha_m = K_\alpha \left(\alpha - \frac{X_\alpha}{V} q + \frac{Y_\alpha}{V} p \right) \quad (57)$$

$$q_m = q \quad (58)$$

$$\theta_m = \theta \quad (59)$$

and

$$N_{Zm} = C_N \frac{\bar{q}S}{mg} + \frac{X_{An}}{g}\dot{q} + \frac{Z_{An}}{g}(q^2+p^2) - \frac{Y_{An}}{g}\dot{p} \quad (60)$$

The K_α term in equation (57) corrects for boom bending and upwash. Since these models were not available for the flight test aircraft, this term is deleted. The assumptions made above also allow deletion of the lateral terms, yielding a set of measurement equations of the form

$$\alpha_m = \alpha - \frac{X_\alpha}{V} q \quad (61)$$

$$q_m = q \tag{62}$$

$$\theta_m = \theta \tag{63}$$

and

$$N_{Zm} = (C_{No} + C_{N\alpha}\alpha + C_{N\delta e}\delta e)\frac{\bar{q}S}{mg} + \frac{X_{An}}{g}\dot{q} + \frac{Z_{An}}{g}q^2 \tag{64}$$

where the C_N term is expanded consistent with equation (54). These equations are used primarily to correct measured data for offsets of the respective sensors (angle of attack and acceleration packages) from the center of gravity of the aircraft. In fact, the terms X_α, X_{An}, and Z_{An} are those offsets from the center of gravity to either the angle of attack vane or the normal acceleration measurement assembly. In the Cyber MMLE3 program, these equations are incorporated into the observation matrices in the model. However, to ease processing required every time the model is accessed, the calculations are made to correct the measured data prior to importing it into the program. This simplifies the observation matrices in the model significantly.

If parameters for the MMLE model are defined as

$$p(1) = C_{N\alpha}$$

$$p(2) = C_{M\alpha}$$

$$p(3) = C_{Mq}$$

$$p(4) = C_{N\delta c}$$

$$p(5) = C_{M\delta c}$$

the state space model is represented by the matrices

$$A = \begin{bmatrix} -\frac{\bar{q}S}{mV\cos\alpha_1}p(1) & 1 & 0 \\ \frac{\bar{q}Sc}{I_{yy}}p(2) & \frac{\bar{q}Sc^2}{2VI_{yy}}p(3) & 0 \\ 0 & \cos\phi_1 & 0 \end{bmatrix} \quad (65)$$

$$B = \begin{bmatrix} -\frac{\bar{q}S}{mV\cos\alpha_1}p(4) & \frac{g\cos\theta_1\cos\phi_1}{V\cos\alpha_1} - \frac{\bar{q}S}{mV\cos\alpha_1}C_{No} \\ \frac{\bar{q}Sc}{I_{yy}}p(5) & \frac{\bar{q}Sc}{I_{yy}}C_{Mo} \\ 0 & \dot{\theta}_0 \end{bmatrix} \quad (66)$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \frac{\bar{q}S}{mg}p(1) & 0 & 0 \end{bmatrix} \quad (67)$$

and

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{\bar{q}S}{mg}p(4) & \frac{\bar{q}S}{mg}C_{No} \end{bmatrix} \quad (68)$$

This is the final form of the model which is coded for using MMLE on flight test data. The model requires quite a bit more information to run than the previous short period approximation and transfer function models. Subsequently, a macro file was created to provide a user interface in the PC Matlab environment to run MMLE. The resulting macro file, called HSINIT.M, is shown in Figure 7 below.

The initialization macro file, when executed, prompts the user to enter required data interactively. The required inputs include physical aircraft parameters, wind tunnel estimates of the five stability derivatives, and the location of the flight test data file. In particular, the required aircraft parameters are

1. **sref** = aircraft reference area (S) in square feet.
2. **cbar** = mean aerodynamic chord in feet.
3. **gw** = aircraft gross weight in pounds.
4. **cg** = aircraft center of gravity in percent of mean aerodynamic chord.
5. **iyy** = pitch moment of inertia in slug-square feet.
6. **xab** = the x offset (positive forward) of the angle of attack vane from the center of gravity in feet.
7. **xan** = the x offset (positive forward) of the normal accelerometer from the center of gravity in feet.

```
clear;
%
% INITIAL SETUP MACRO FOR RUNNING HSMMLE.M
% MUST BE RUN PRIOR TO HSMMLE.M IN ORDER TO
% INPUT REQUIRED CONSTANTS, WIND TUNNEL
% ESTIMATES, AND LOAD THE FLIGHT TEST DATA
%
%------------------------------- INPUT REQUIRED CONSTANTS
%
sref=input('REFERENCE AREA (S) IN SQUARE FEET? ');
cbar=input('MEAN AERODYNAMIC CHORD IN FEET? ');
gw=input('AIRCRAFT GROSS WEIGHT IN POUNDS? ');
cg=input('AIRCRAFT CG IN %MAC? ');
iyy=input('MOMENT OF INERTIA (IYY) IN SLUG-FT^2? ');
xab=input('X-DIST IN FT FROM CG TO ALPHA VANE (+FWD)? ');
xan=input('X-DIST IN FT FROM CG TO NZ ACCEL (+FWD)? ');
zan=input('Z-DIST IN FT FROM CG TO NZ ACCEL (+DWN)? ');
vtrue=input('TRUE AIRSPEED IN FEET PER SECOND? ');
qbar=input('DYNAMIC PRESSURE IN LBS PER SQUARE FOOT? ');
%
%---------------------------- INPUT WIND TUNNEL ESTIMATES
%
pref(1)=input('CN_ALPHA FROM WIND TUNNEL (1/DEG)? ');
pref(2)=input('CM_ALPHA FROM WIND TUNNEL (1/DEG)? ');
pref(3)=input('CM_Q FROM WIND TUNNEL (1/RAD)? ');
pref(4)=input('CN_DE FROM WIND TUNNEL (1/DEG)? ');
pref(5)=input('CM_DE FROM WIND TUNNEL (1/DEG)? ');
pref=pref*180/pi;
pref(3)=pref(3)*pi/180;
%
%---------------------------------------- LOAD DATA FILE
%
% A FLIGHT TEST DATA FILE MUST EXIST AS AN ASCII FILE
% ON THE MATLAB DIRECTORY (WITH A .DAT EXTENSION)
%
data=input('DATA FILE (WITHOUT .DAT EXTENSION)? ','s');
ndp=input('NUMBER OF DATA POINTS? ');
ldc=['load ',data,'.dat;'];
eval(ldc);
%
dt=1/32;
t=[0:ndp-1]*dt;
%
save hsinit
%
%-------------------------------------------ENDHSINIT.M
```

Figure 7. MMLE Initialization Macro Code

8. **zan** = the z offset (positive down) of the normal accelerometer from the center of gravity in feet.
9. **vtrue** = true airspeed in feet per second.
10. **qbar** = dynamic pressure in pounds per square foot.

The wind tunnel values for the stability derivatives are input in common units of per degree for all but C_{Mq}, which is per radian. The macro then converts all units to radians to avoid dimensional problems in the model. The flight test data must exist in an ASCII file on the PC Matlab directory. The macro is set up to import an ASCII file with a .dat file extension. Finally, the macro creates a time vector with the 32 samples per second which are available on the flight test aircraft, as well as the user input number of data points in the data sample. The initial data is then saved in a data file to be accessed by the MMLE macro file.

The macro file HSP2SS.M performs the parameter-to-state-space conversion. The file is set up to implement the matrices shown in equations (65) through (68). HSP2SS.M is shown in Figure 8. The macro calculates the bias terms in these equations such that the rates of the state variables are zeroed, and the load factor measurement is set to its initial value. The remainder of the code is identical to the parameter-to-state-space macros described in the simulation section.

```
function[a,phi,gam,c,d,q,x0,dt,rowinq,b]=hsp2ss(p);
%-------------------------------------------------------
% P2SS FUNCTION FOR HSMMLE.M ---- EXTENSIVELY COMMENTED
%-------------------------------------------------------
%
% p(1) = CN_ALPHA    p(2) = CM_ALPHA      p(3) = CM_Q
% p(4) = CN_DE       p(5) = CM_DE
%
%---------------------------- PERFORM INITIAL CALCULATIONS
cnst1=(-1*qbar*sref)/(cos(al1)*gw*vtrue/32.17);
cnst2=qbar*sref*cbar/iyy;
cnst3=cnst2*cbar/(2*vtrue);
cnst4=32.17*cos(ph1)*cos(th1)/(vtrue*cos(al1));
cnst5=qbar*sref/gw;
cnst6=-1*(cnst1*p(1)*al1+q1+cnst1*p(4)*de1+cnst4);
cnst7=-1*(cnst2*p(2)*al1+cnst3*p(3)*q1+cnst2*p(5)*de1);
cnst8=-1*(q1*cos(ph1));
cnst9=(-1*(cnst5*p(1)*al1+cnst5*p(4)*de1))+nz1;
%-----------------------------------STABILITY DERIVATIVES
a=[cnst1*p(1)              1                  0
   cnst2*p(2)          cnst3*p(3)             0
       0                cos(ph1)              0];
%-----------------------------------CONTROL DERIVATIVES
b=[cnst1*p(4)        cnst4+cnst6
   cnst2*p(5)           cnst7
       0                cnst8      ];
%-------------------------------------MEASUREMENT MATRIX
c=[    1                   0                   0
       0                   1                   0
       0                   0                   1
   cnst5*p(1)              0                   0   ];
%------------------------------------ FEED THROUGH MATRIX
d=[    0                   0
       0                   0
       0                   0
   cnst5*p(4)           cnst9   ];
%---------------------------------- STATE NOISE COVARIANCE
q=eye(3)*1e-6;%    SAME SIZE AS A WITH Q*Q' POS. DEFINITE!
%--------------------- ROWS IN Q IN WHICH PARAMETERS OCCUR
rowinq=0*p;
%------------------------------------ INITIAL STATE VECTOR
x0=[al1 q1 th1];
%------------------------------- DEFINE DT AND DISCRETIZE
dt=1/32;
[phi,gam]=c2d(a,b,dt);
%-----------------------------------------------END HSP2SS.M
```

Figure 8. Flight Test P2SS Function Code

All the required up front calculations are located in the HSMMLE.M macro file (shown in Figure 9). The macro first loads in the file containing the results of the HSINIT.M run. Then, the specific inputs and outputs are selected from the flight test data file. In this particular case, there is a correction of approximately four degrees for a known angle of attack calibration offset. The data are also converted to radians for unit consistency in the program. Angle of attack and load factor measurements are corrected for the distance of the sensors from the center of gravity, and initial values for the states, measurements, and control inputs are calculated from the flight test data file. The parameter identification options are then set. The only difference in these setting from the simulation is that the initial parameter guess is set equal to the wind tunnel estimates. The innovations covariance matrix is set to the default for MMLE3, which has been developed over the years as a good rule of thumb for flight test data (5:12). The last section of the macro file performs final calculations to convert the stability derivative estimates back into common units, calculate equivalent second-order system parameters, and display the outputs. The second-order system damping ratio and natural frequency are calculated using the final system A matrix and the PC Matlab DAMP function. The DAMP function returns these parameters

```
format compact,clc
clear;
load hsinit;
!erase hsmmle.log;
diary hsmmle.log
'-------------------------------------------------------';
disp('HAVE STABILITY PARAMETER IDENTIFICATION MACRO FILE')
disp(' T-38A LONGITUDINAL STAB. AND CONTROL DERIVATIVES')
disp('              ONE INPUT, FOUR OUTPUTS')
global sref cbar gw iyy vtrue qbar al1 th1 ph1 q1 nz1 de1;
%
%-------------------------------- INPUT FLIGHT TEST DATA
uydata=zeros(ndp,6);
col1=['uydata(:,1)=',data,'(:,7);'];
eval(col1);
uydata(:,1)=uydata(:,1)*pi/180;
uydata(:,2)=ones(ndp,1);
col3=['uydata(:,3)=',data,'(:,9)+(4.07*ones(ndp,1));'];
eval(col3);
uydata(:,3)=uydata(:,3)*pi/180;
col4=['uydata(:,4)=',data,'(:,8);'];
eval(col4);
uydata(:,4)=uydata(:,4)*pi/180;
col5=['uydata(:,5)=',data,'(:,10);'];
eval(col5);
uydata(:,5)=uydata(:,5)*pi/180;
col6=['uydata(:,6)=',data,'(:,12);'];
eval(col6);
%
% CORRECT ALPHA AND NZ FOR DISTANCE OF SENSORS FROM CG
uydata(:,3)=uydata(:,3)+(xab*uydata(:,4)/vtrue);
for i=1:ndp,
     q2t(i)=uydata(i,4)*uydata(i,4);
     q2=q2t';
end
uydata(:,6)=uydata(:,6)-(zan*q2/32.17);
qdot=[diff(uydata(:,4))*32.0;0];
uydata(:,6)=uydata(:,6)-(xan*qdot/32.17);
% INITIAL CONDITIONS FOR ALPHA, THETA, PHI, Q, NZ, AND DE
al1=uydata(1,3);
th1=uydata(1,5);
ph1ic=['ph1=',data,'(1,11)*pi/180;'];
eval(ph1ic);
q1=uydata(1,4);
nz1=uydata(1,6);
de1=uydata(1,1);
```

Figure 9. Flight Test MMLE Macro Code (Sheet 1 of 2)

```
%------------------------ SUFFICIENT INPUTS TO MMLE FOLLOW
pidq=[1];%  IDENTIFY WHICH PARAMETERS ARE TO BE IDENTIFIED
pidm=[1];%  IN THE QUADRATIC, MARQUARDT, AND FINAL STAGES.
pidf=[1:5];%pidq, pidm, pidf MUST BE VALID IF NOT USED
p0=pref;%  INITIAL PARAMETER GUESS = WIND TUNNEL ESTIMATES
opt=[0  0 10  0 .02 .001 .001 1];%  MAXIMUM ITERATIONS AND
%                                        CONVERGENCE CRITERIA
radc=180/pi;
ggiv=[10*(radc^2) 60*(radc^2) 30*(radc^2) 200];
gg0=inv(diag(ggiv));%    INNOVATIONS COV. (MMLE3 DEFAULT)
p2snam='hsp2ss'; %              FILENAME OF P2SS FUNCTION
pert=1e-4; %    PERTURBATION USED FOR GRADIENT CALCULATION
linesearch=1;
mmle
%------------------------------ PERFORM FINAL CALCULATIONS
cna=pfin(1)/radc;
cma=pfin(2)/radc;
cmq=pfin(3);
cnde=pfin(4)/radc;
cmde=pfin(5)/radc;
constf=sqrt(qbar*sref*cbar/iyy);
[wn,zeta]=damp(a);
ktheta=cmde*radc*qbar*sref*cbar/iyy;
invth2=cna*radc*32.17*qbar*sref/(vtrue*gw);
disp(' ')
disp('  DATA FILE = '),disp(data)
disp(' ')
cond=[cg al1*180/pi qbar];
disp('     CG      TRIM AOA   QBAR')
disp(' ')
disp(cond)
disp(' ')
disp(' ')
deriv=[cna cma cmq cnde cmde];
disp('     CNA       CMA       CMQ       CNDE      CMDE')
disp(' ')
disp(deriv)
disp(' ')
disp(' ')
system=[zeta(3) wn(3) ktheta invth2];
disp('     ZETA       WN      KTHETA   1/T_THETA2')
disp(' ')
disp(system)
diary off
!print hsmmle.log;
%-----------------------------------------ENDHSMMLE.M
```

Figure 9. Flight Test MMLE Macro Code (Sheet 2 of 2)

for the input A matrix. The system parameters K_θ and $1/T_{\theta 2}$ are calculated as

$$K_\theta \approx M_{\delta e} \tag{69}$$

and

$$\frac{1}{T_{\theta 2}} \approx N_\alpha \tag{70}$$

The end result of the HSINIT.M, HSP2SS.M, and HSMMLE.M macro files is a user-friendly, interactive capability to perform parameter estimation from flight test data using a personal computer. The ultimate test of the system, however, is to accomplish a flight test program to verify the results. The following section will describe the details of the flight test planning, execution, and results.

V. Flight Test

Introduction

This section presents the results of a limited development and evaluation of a parameter estimation capability for the USAF Test Pilot School (USAFTPS), using the final model presented in the last section. To assess this capability, flight test data were gathered in a T-38A aircraft.

Several past attempts have been made to demonstrate parameter estimation capability at the USAFTPS. The most recent was the T-38 Parameter Estimation Study (HAVE PEST), which was conducted from September through December of 1989. These past projects suffered from problems with both the data acquisition system and computer software availability. Appropriate steps were taken to resolve these problems for this test.

The test team flew 9 test sorties and 3 support sorties, all in the T-38A, for a total of 12.9 flight hours. The test missions were flown from 17 September through 19 October 1990 at the AFFTC, Edwards AFB, California.

The overall test objective was to evaluate the parameter estimation capability for the USAFTPS, concentrating only on the longitudinal modes of a T-38A aircraft.

The specific objectives were to:

1. Evaluate the personal computer based approach to aircraft parameter estimation using the MMLE routine in PC Matlab. Specifically, to demonstrate the capability to process data from the USAFTPS data reduction facility through these programs to accomplish the parameter estimation task.

2. Compare results obtained using this method available at USAFTPS with results of the MMLE3 program run on the Ridley Mission Control Center (RMCC) Cyber computer.

3. Compare equivalent system parameters obtained from the MMLE results to those obtained from frequency response analysis (FRA) methods available on the RMCC Cyber computer, as well as from classical modal analysis flight test techniques.

Test Item Description

The personal computer hardware used for this test was a Dell 386/25 with a 387 Math Coprocessor. The primary software used was the PC Matlab program.

The PC Matlab program was a matrix operation program which could be used to solve practical engineering and mathematical problems. PC Matlab was designed for specific applications such as control theory and signal processing. (8:3). Two additional "toolboxes" were used for this

project: the Control System Toolbox (6) and the State Space Identification Tool (10).

Flight test data from the T-38A were used as a tool to evaluate the parameter estimation capabilities of the USAFTPS. The test aircraft, USAF serial number (S/N) 63-8135, was a production aircraft modified with the flight test instrumentation described below. The T-38A, built by Northrop Corporation, was a supersonic, two-place, tandem trainer powered by two afterburning J85-GE-5 series turbojet engines. The flight controls were powered by a conventional, irreversible hydraulic system. The aircraft was flown with no external stores. More information about the production model T-38A can be found in the T-38 Flight Manual (17).

The test aircraft was modified as follows:

1. Nose boom with a Yaw, Angle-of-attack, Pitot-static System (YAPS) head.
2. Metraplex airborne test instrumentation system with 32 samples per second data collection rate.
3. Sensitive airspeed, Mach, and G indicators.
4. C-band beacon.

Instrumentation

Table A1 in Appendix A shows the instrumentation parameters and available resolutions for this test. The

Metraplex system was essential to the success of the test program. Its 32 sample per second sampling rate and greater parameter resolution were a must for the parameter estimation process. In general, instrumentation issues were critical to achieving test objectives. Several key lessons were learned regarding instrumentation, and these are summarized below.

Calibrations. A key to the success of the program was ensuring that the instrumentation was properly calibrated. To accomplish this, the test team developed specific calibration requirements (including calibration ranges and increments). To ensure the quality of the calibrations, test team members participated actively in the entire calibration process. Critical parameters (angle of attack, pitch rate, normal acceleration, and elevator deflection) were calibrated three times to minimize errors. In some cases, the calibration curves were nonlinear at the extremes of the calibration ranges. The USAFTPS data reduction system could only implement first order (i.e. linear) curve fits. In those instances where nonlinear calibrations existed, the ranges of the calibrations were reduced to accurately represent the regions of primary interest.

Parameter Resolution. Resolution of key parameters was also critical to the success of the parameter estimation effort. Based on the data acquisition system configuration,

calibration ranges were narrowed to achieve desired resolutions. Even so, the pitch angle resolution turned out to be inadequate. Better resolution could have been achieved by narrowing the calibration range even further.

Parameter Filtering. Data from four otherwise effective sorties were corrupted by a filter mismatch among the key longitudinal parameters. The data acquisition system was set up to filter some parameters with 10 Hertz filters, some with 20 Hertz filters, and some not at all. This created a problem with frequency response analysis techniques, and to a limited degree with the MMLE techniques. Filtering of key parameters should be checked for consistency to avoid inducing unwanted phase lag differences between parameters.

Other Issues. Several inaccuracies were introduced which caused instrumentation measurements to differ from true aircraft response. The positions of accelerometer packages and nose boom measurement vanes relative to aircraft center of gravity directly affected acceleration and angle measurements from those sources. In fact, vertical displacements could not be determined due to uncertainty of the vertical datum for the aircraft. Accurate measurements of these relative positions are a must for getting accurate parameter estimation. In addition, corrections could have been made for boom bending and upwash

in angle of attack measurements. No such models were readily available, and these corrections were not made for this test. More accurate parameter estimates could be obtained by including such a model.

Test and Evaluation

Test Procedures. Data were collected for classical analysis using pitch doublets at each test point. Figure A1 in Appendix A shows the four test points and their positions in the T-38A operational flight envelope. Table A2 describes each test point in detail. Doublets were performed at approximately the short period natural frequency of the aircraft. These maneuvers and the corresponding hand reduction of the data are described in the Flying Qualities Phase Textbook (1:8.47-8.53).

Flight test data were gathered for MMLE techniques using pitch doublets at each test point. Doublets were repeated at elevated angles of attack (in increments to give at least two test points above trim without encountering buffet). Load factor was increased to provide the increase in angle of attack, with altitude loss accepted to maintain Mach number. Doublets were performed with a period of one second. Amplitude was adjusted to get a pitch rate of approximately ±10 degrees per second. Angle of attack perturbations were kept within ±3 degrees. These maneuvers are described in Flight Investigation of Various Control

Inputs Intended for Parameter Estimation (15). Each pilot practiced the MMLE inputs in a ground simulator prior to the start of flight test. Data were processed through the PC Matlab and Cyber MMLE programs, which were discussed in detail in the previous sections. The programs were used to generate stability derivatives, as well as estimated second-order system parameters based on these derivatives. The required aircraft gross weight and moment of inertia data were obtained from Calculated Dead Weight Distribution and Moment of Inertia Report, T-38A (13). The correlation of this data to the actual flight test aircraft was in question, but it was the best data available. Future attempts to perform parameter estimation on the T-38A should calculate these data more precisely for the test aircraft. Wind tunnel estimates were used as starting points for the derivative estimation processes, and were obtained from T-38 Static Stability and Basic Aerodynamic Data (2).

Pitch frequency sweeps were used to generate data for frequency response analysis. Frequency sweeps were performed from approximately 0.2 to 20 radians per second, with the entire sweep taking at least 32 seconds. Each pilot practiced the frequency sweep inputs in a ground simulator prior to the start of flight test. Data were processed through the Cyber FRA program. The FRA program converts time history data into the frequency domain using a

Fourier transform, and produces output frequency, magnitude, and phase angle information (16:826). This information was matched using an equivalent system curve fit, in order to obtain equivalent second-order system parameters. The results were used to compare against the same parameters calculated from the MMLE stability derivative estimates. This provided a means of checking the MMLE results using a completely different (frequency domain) technique.

The frequency sweeps were difficult to perform well. The technique required a very low frequency input initially (about 0.2 radians per second). Pilots found it challenging to make such a slow input and keep the amplitude small enough to avoid large excursions in altitude and airspeed. A smooth increase in frequency was then required, with input amplitude adjusted to keep essentially a constant output amplitude. At the higher frequencies, pilots had to use some outside reference to decouple themselves from the response of the aircraft. Practice helped the quality of the inputs significantly, and pilots generally felt that the time spent in the ground simulator prior to flight was invaluable. Ground simulator training should be accomplished before collecting inflight data using parameter estimation flight test techniques.

Test Results. The data from the MMLE doublets were successfully processed on the PC. Flight test data were

reduced to engineering units through the Test Pilot School PDP 11/84 computer, as detailed in Data Reduction on the Test Pilot School Data Reduction Computer System (3). The existing plot building routine was used to create data files with the specific time slices for MMLE processing. These files could easily be downloaded to the PC using the file transfer protocol (FTP) through the existing USAFTPS personal computer network. The PC Matlab program was capable of reading in ASCII files such as those created by the plot building program, provided the header lines had been deleted.

Although the processing procedures for MMLE on the PC were quite lengthy, they were easily executed using the macro files discussed in previous sections. The MMLE macro files were set up to be as generic as possible and allow the user to input all required program inputs interactively at the terminal.

Figure 10 below shows an example of the time history matching by PC MMLE for test point 1(a). Figures B1 through B11 in Appendix B show the remaining time history matching results from the PC MMLE. In general, the lack of boom bending and upwash corrections to angle of attack can be seen in the lower quality of the match to the flight test angle of attack data. In two particular cases, Figures B4 and B8, the flight test data for angle of attack were

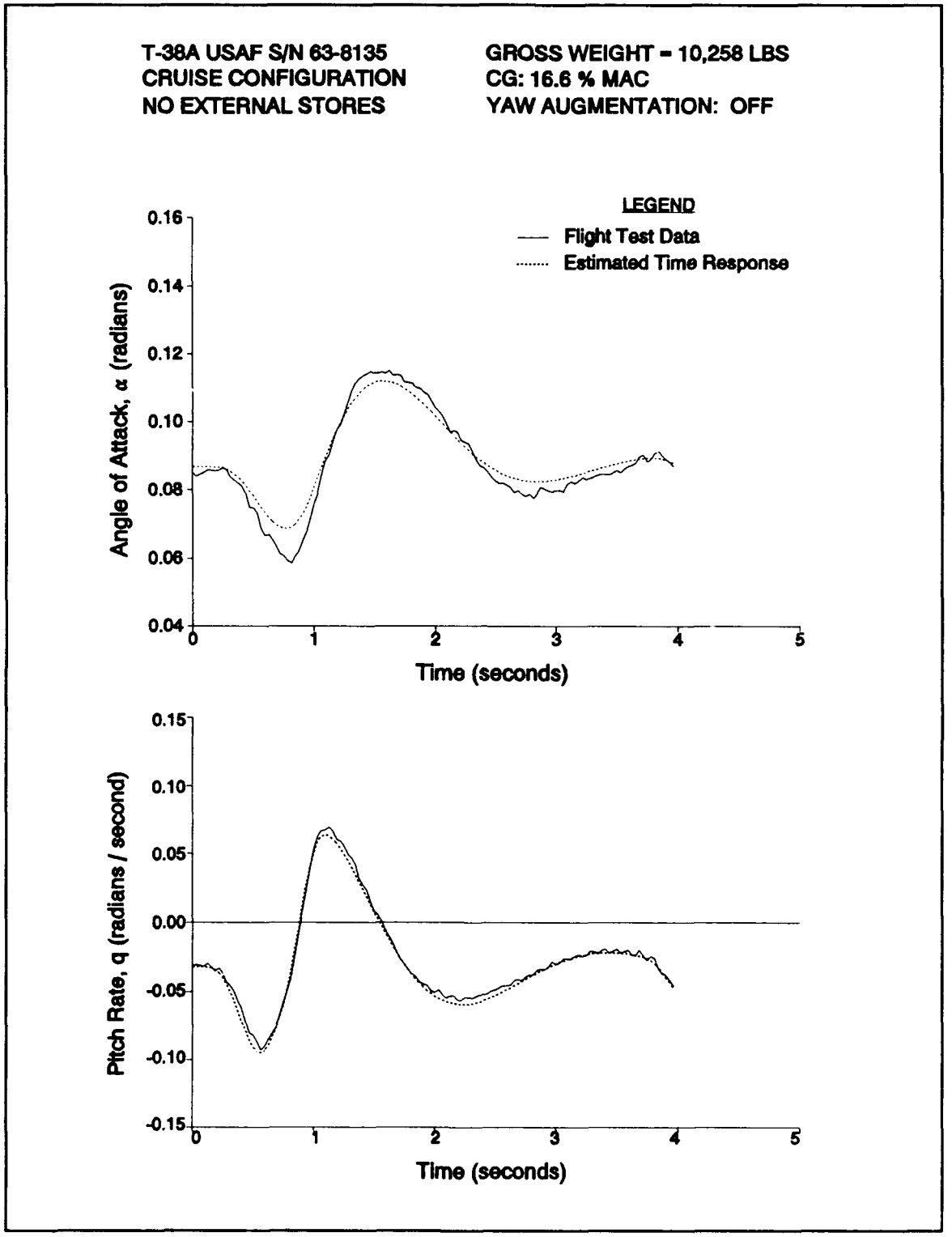

Figure 10. Time History Match, Test Point 1(a) (Sheet 1 of 2)

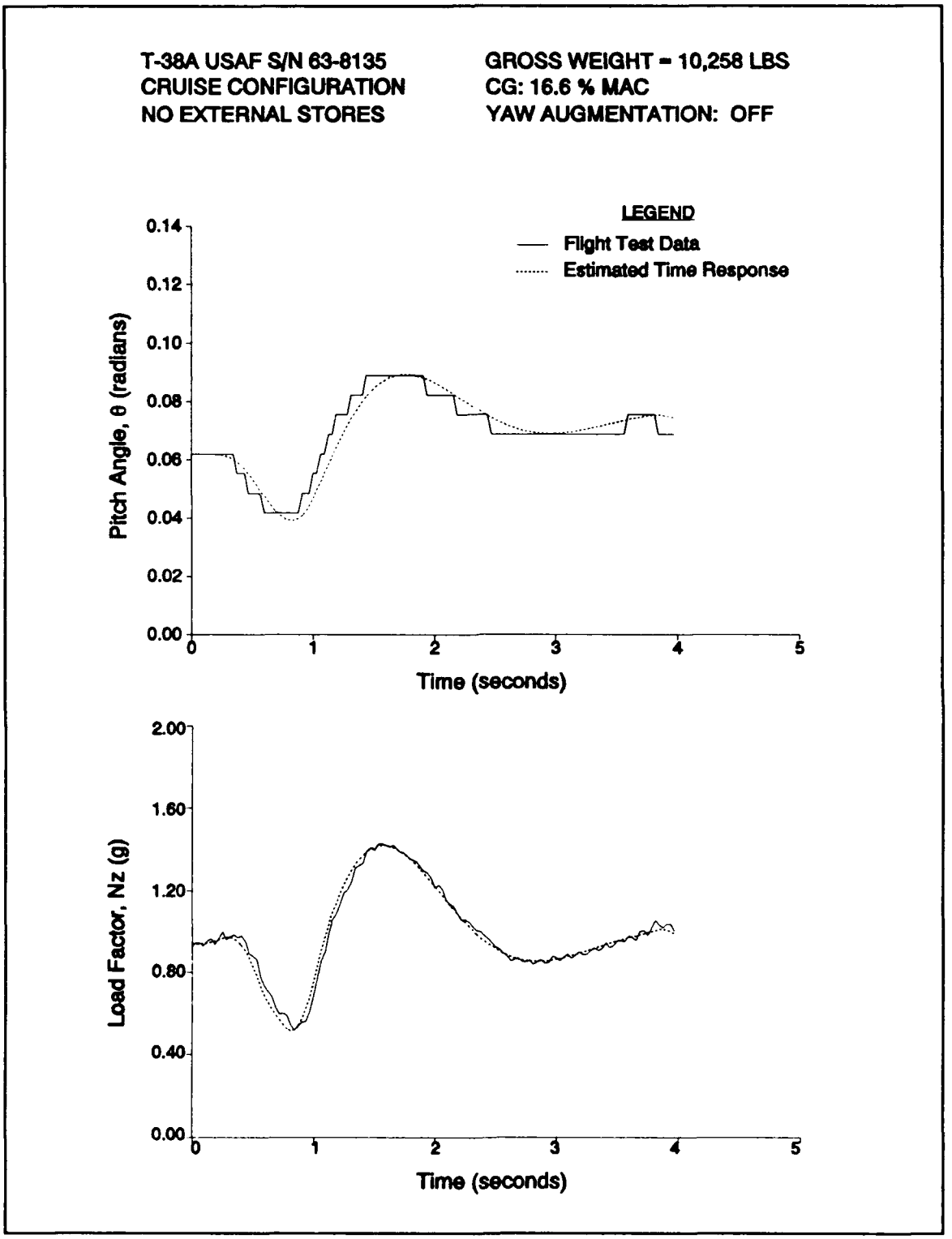

Figure 10. Time History Match, Test Point 1(a) (Sheet 2 of 2)

erroneous. However, the program was still able to complete the parameter identification process with reasonable results.

Figures 11 through 15 on the following pages show the comparison of the results from the PC and the Cyber for MMLE estimation of longitudinal stability derivatives. The derivatives were estimated based on the Test Pilot School sign convention, as shown in Figure A2 in Appendix A. Different symbols are presented for the PC and Cyber results at the two dynamic pressures tested. The vertical bars indicate the modified Cramér-Rao bounds on the Cyber estimates. The Cramér-Rao bounds are multiplied by a factor of ten, which is used to indicate the uncertainty level in the parameter estimates. Recall that the Cramér-Rao bounds are the lower limit on the variance of the estimates, and are frequently multiplied by a scale factor to arrive at uncertainty levels (4:11). These bounds are used in lieu of confidence intervals, and indicate the confidence in the Cyber estimates for each data run. A detailed description of Cramér-Rao bounds can be found in the User's Manual for MMLE3 (7). The results from the PC correlated well with those from the Cyber. In all cases, the PC estimates were within the uncertainty bounds. In fact, the vast majority of the PC runs fell well within the bounds of the Cyber estimates. Overall, the quality of the PC MMLE results

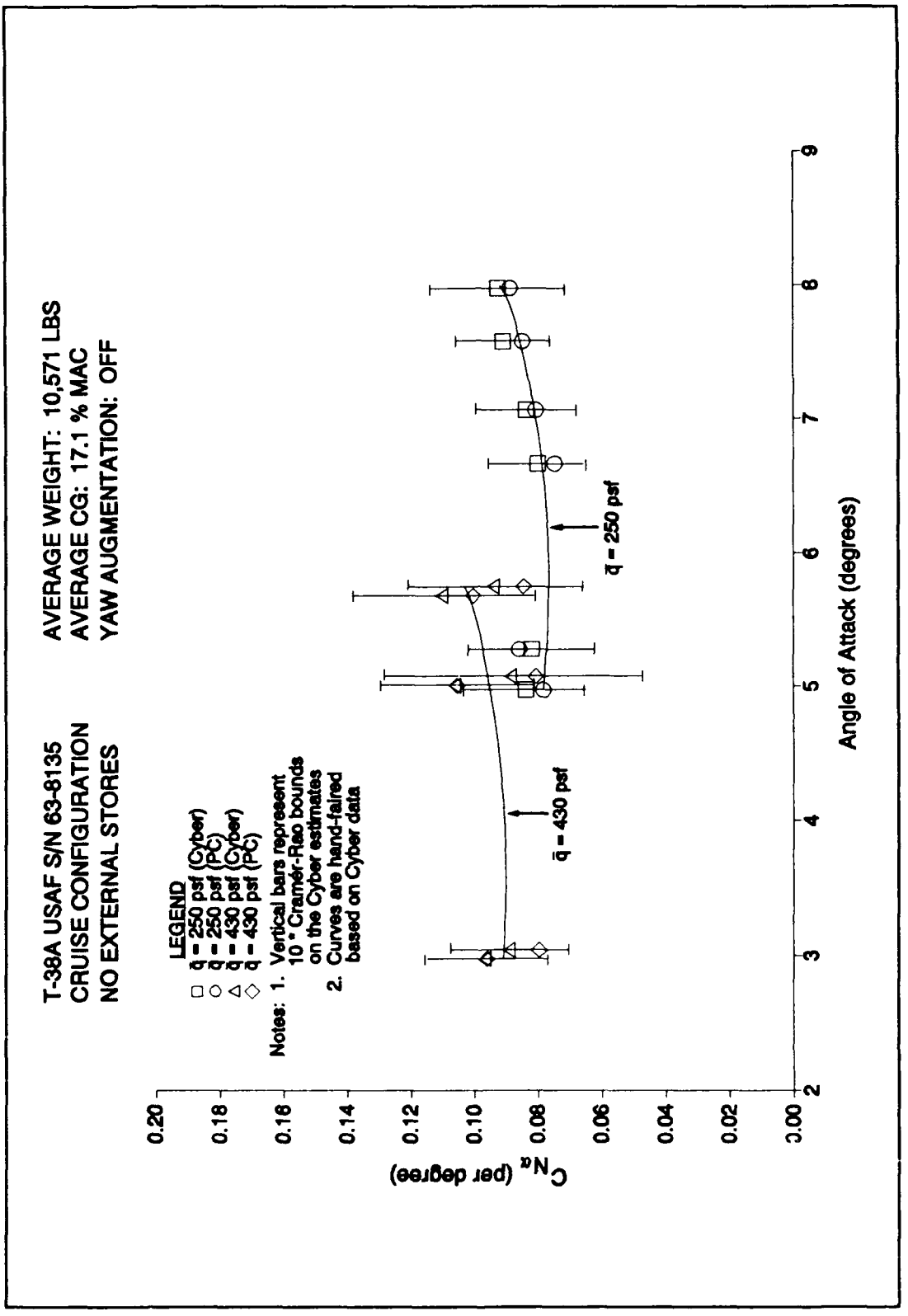

Figure 11. MMLE Summary (C_{N_α})

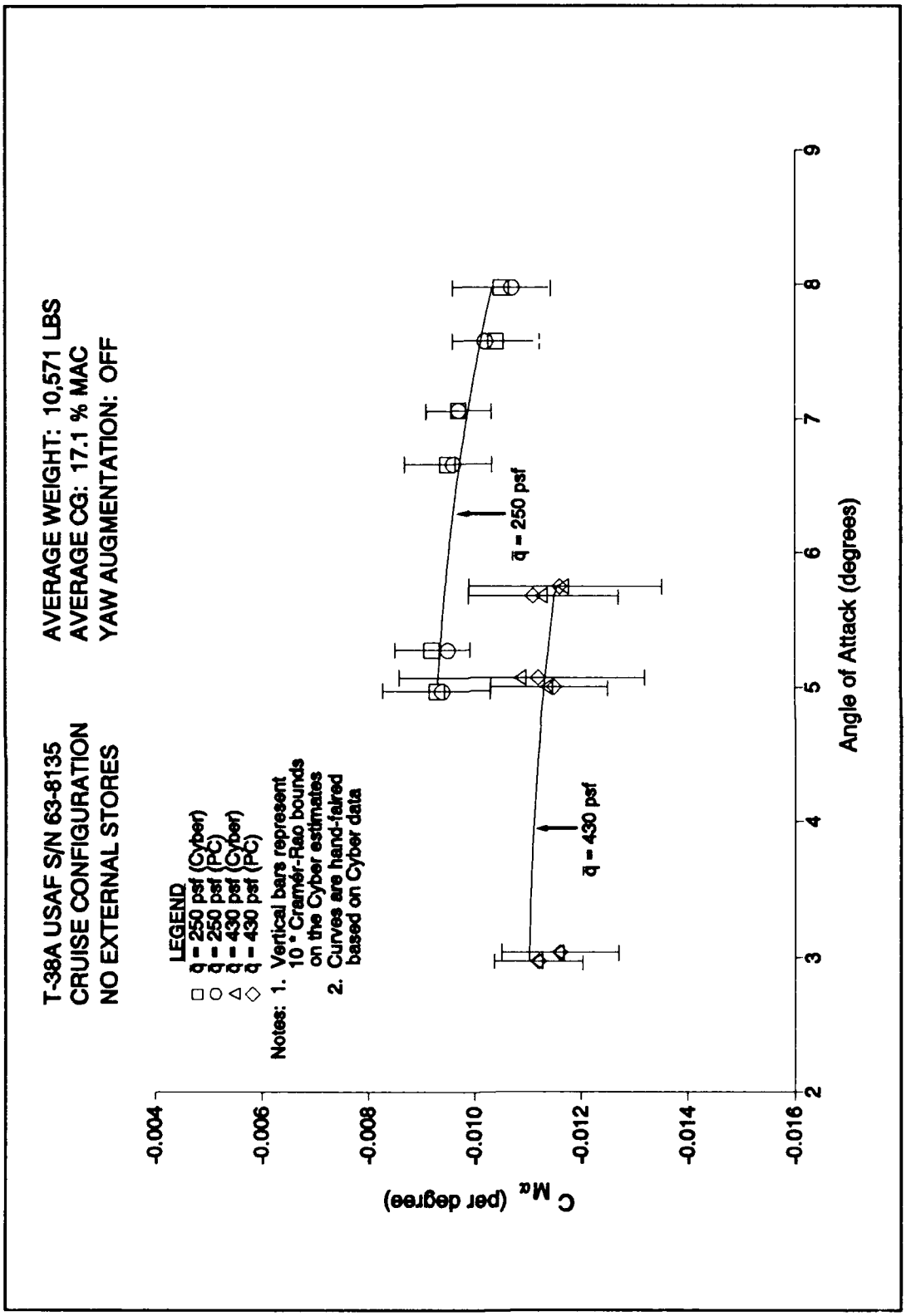

Figure 12. MMLE Summary (C_{M_α})

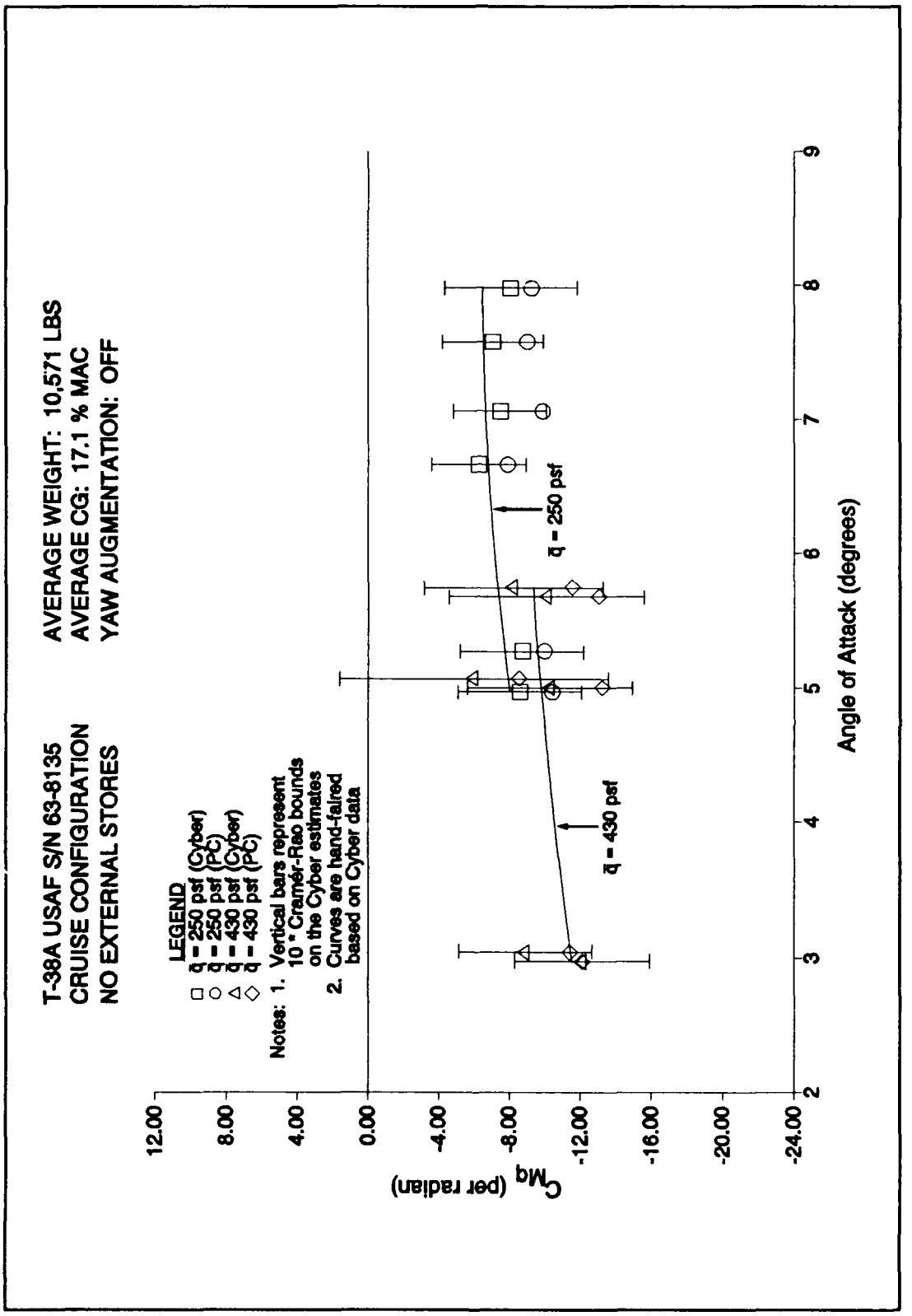

Figure 13. MMLE Summary (C_{Mq})

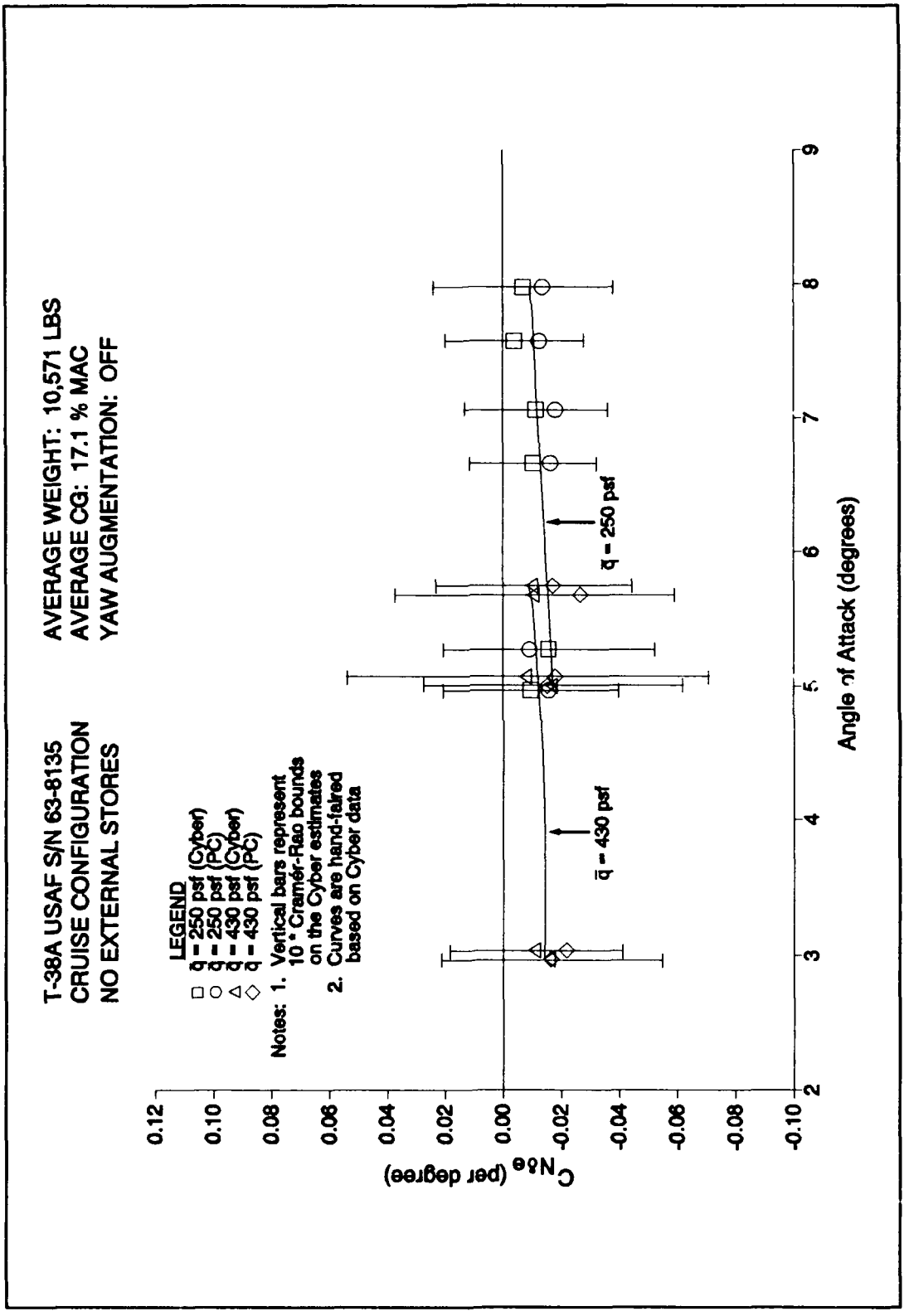

Figure 14. MMLE Summary ($C_{N\delta e}$)

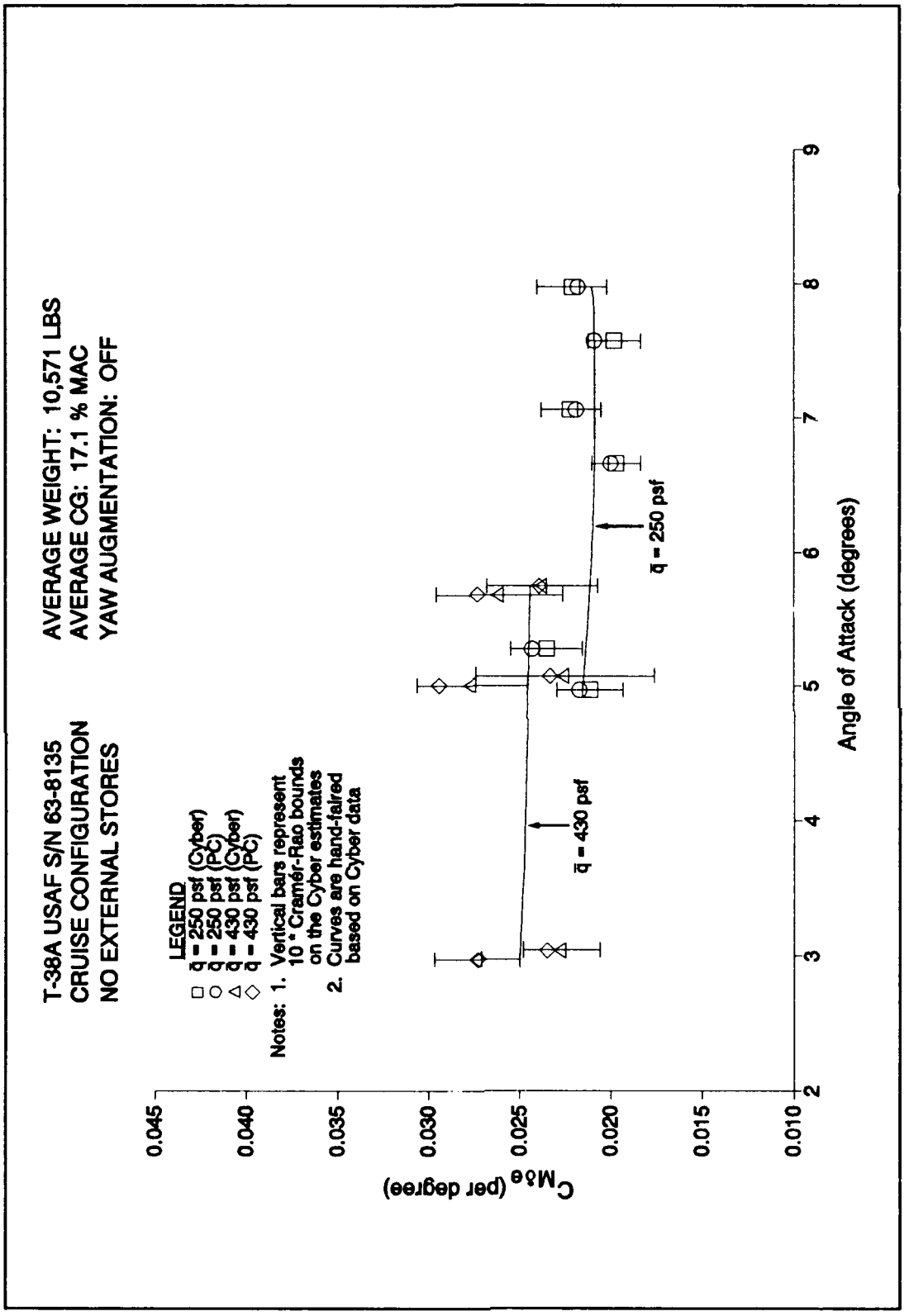

Figure 15. MMLE Summary ($C_{M\delta e}$)

obtained was satisfactory for integration into the Test Pilot School curriculum. However, the scope of the evaluation was limited to longitudinal aircraft responses. The capability of the Test Pilot School to identify lateral-directional responses should be evaluated prior to incorporating MMLE into the flying qualities course.

Figure 16 below shows an example of the Cyber FRA results from test point 1(a). Figures B12 through B14 in Appendix B present the remaining Cyber results. Data from multiple time histories were ensemble averaged to improve the quality of the estimation. This project achieved excellent results from FRA using frequency sweep inputs. The use of ensemble averaging was a key factor in generating such high quality results. Ensemble averaging should be considered a must for future attempts at frequency response analysis. The data are presented in Bode plot form, with magnitude and phase angle of the pitch rate to elevator deflection transfer function plotted versus frequency. The Cyber plots show estimates with 95 percent confidence intervals. In general, the Cyber results were excellent. The confidence intervals were tight out to approximately 15 radians per second. A lower-order equivalent system (LOES) curve fit was used to match the frequency characteristics from Figures 16 and B12 through B14. The matching results are plotted in Figures 17 through 20 on the following pages.

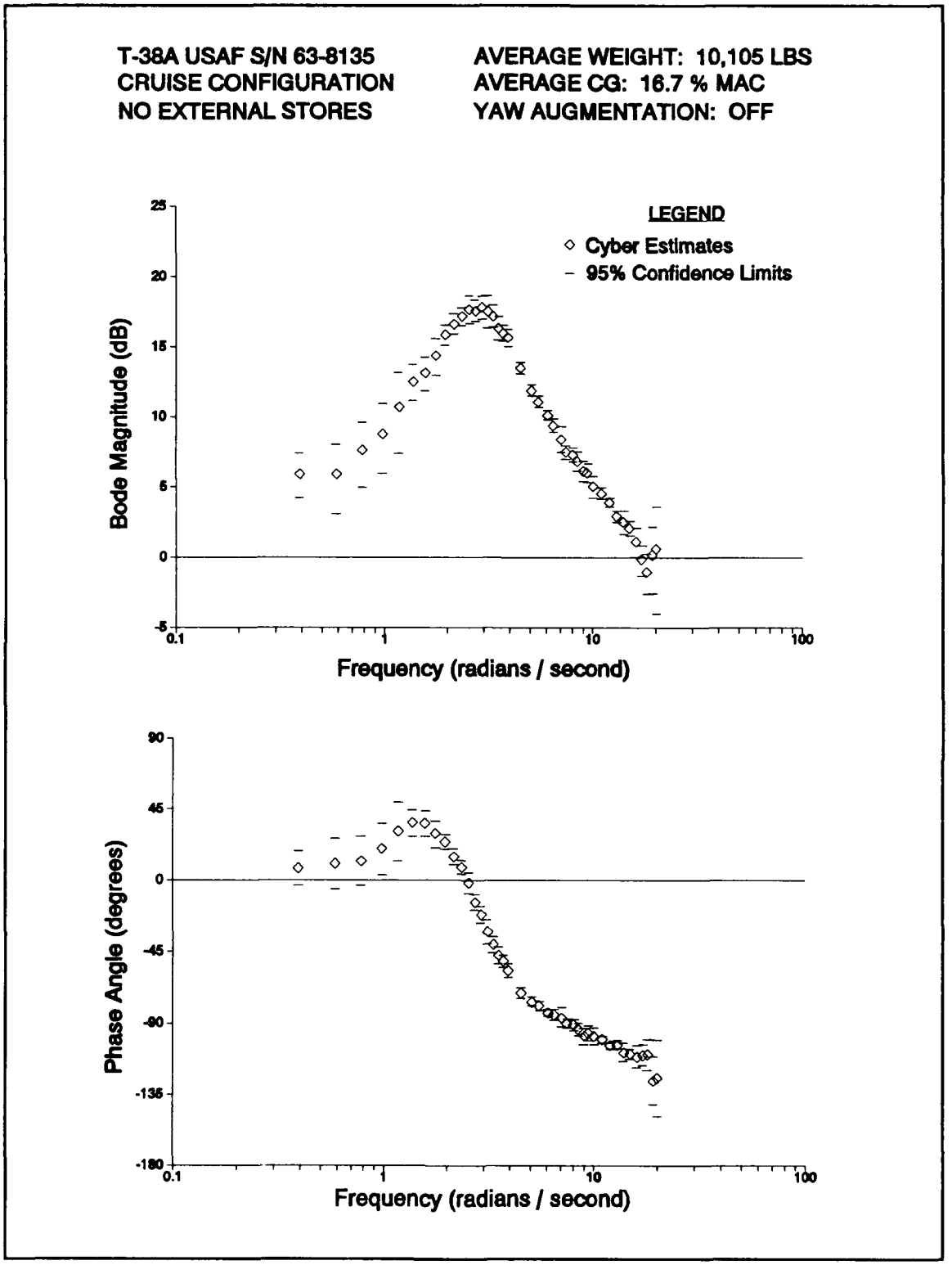

Figure 16. FRA Results, Test Point 1(a)

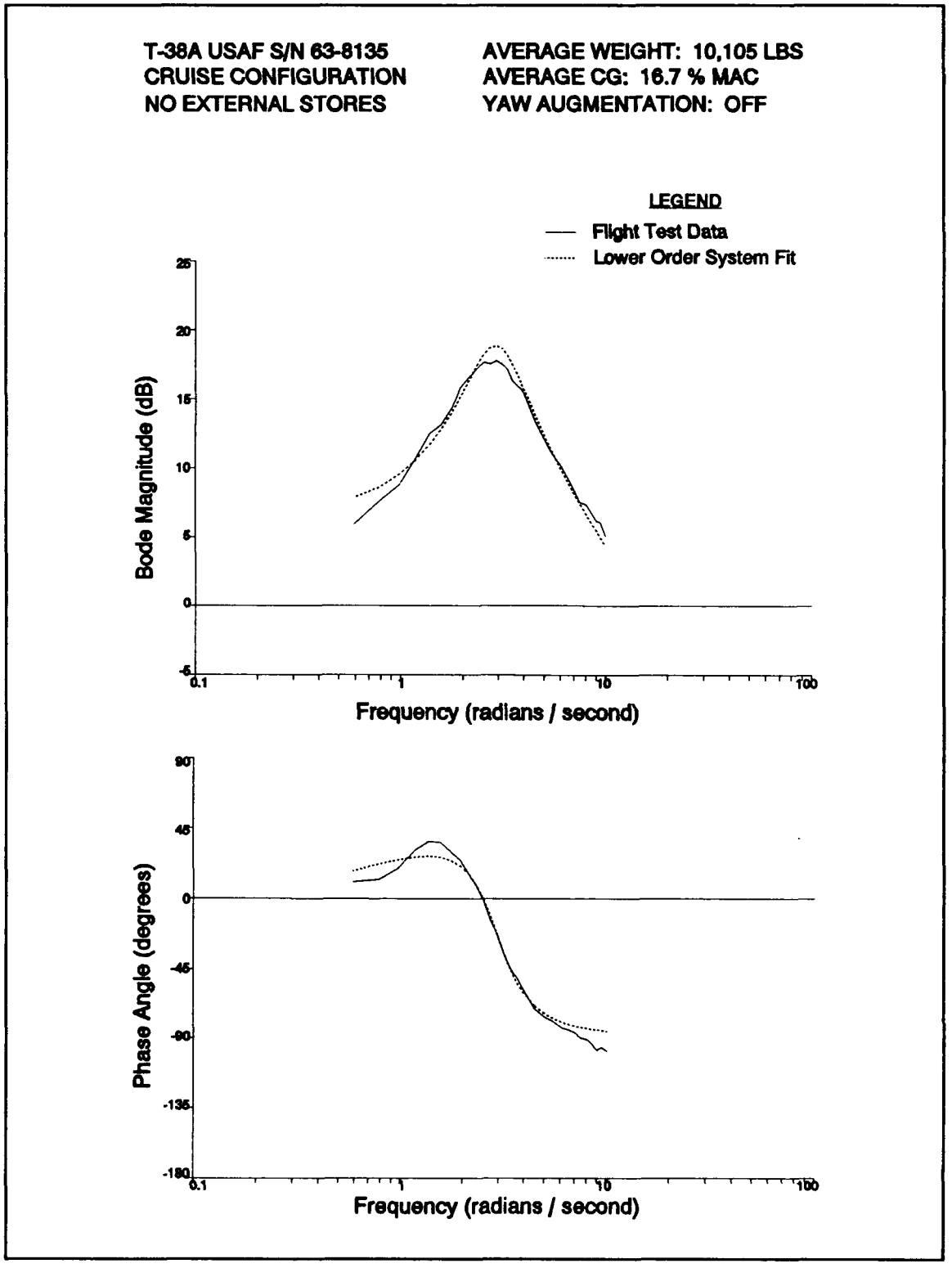

Figure 17. LOES Match, Test Point 1(a)

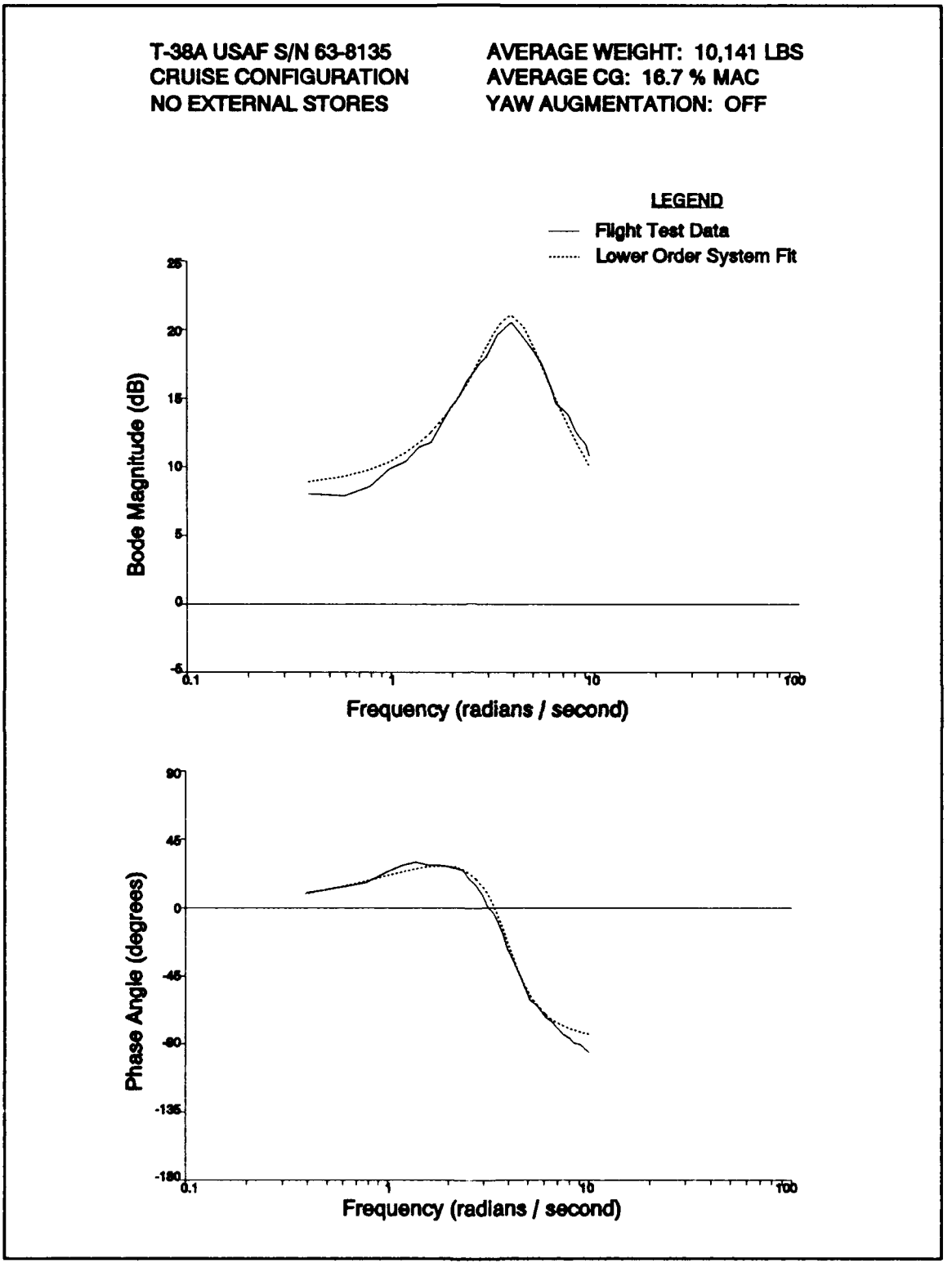

Figure 18. LOES Match, Test Point 2(a)

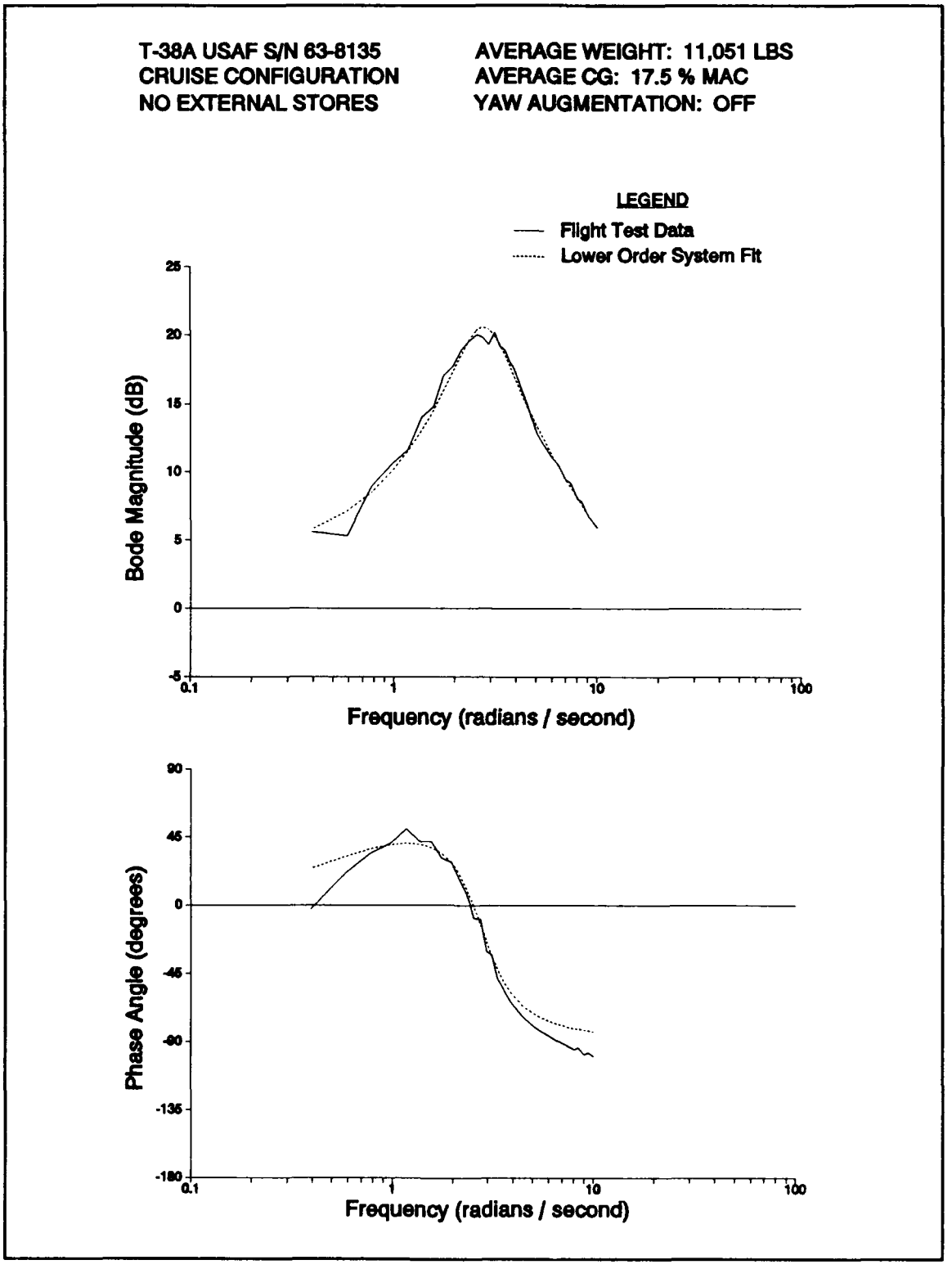

Figure 19. LOES Match, Test Point 3(a)

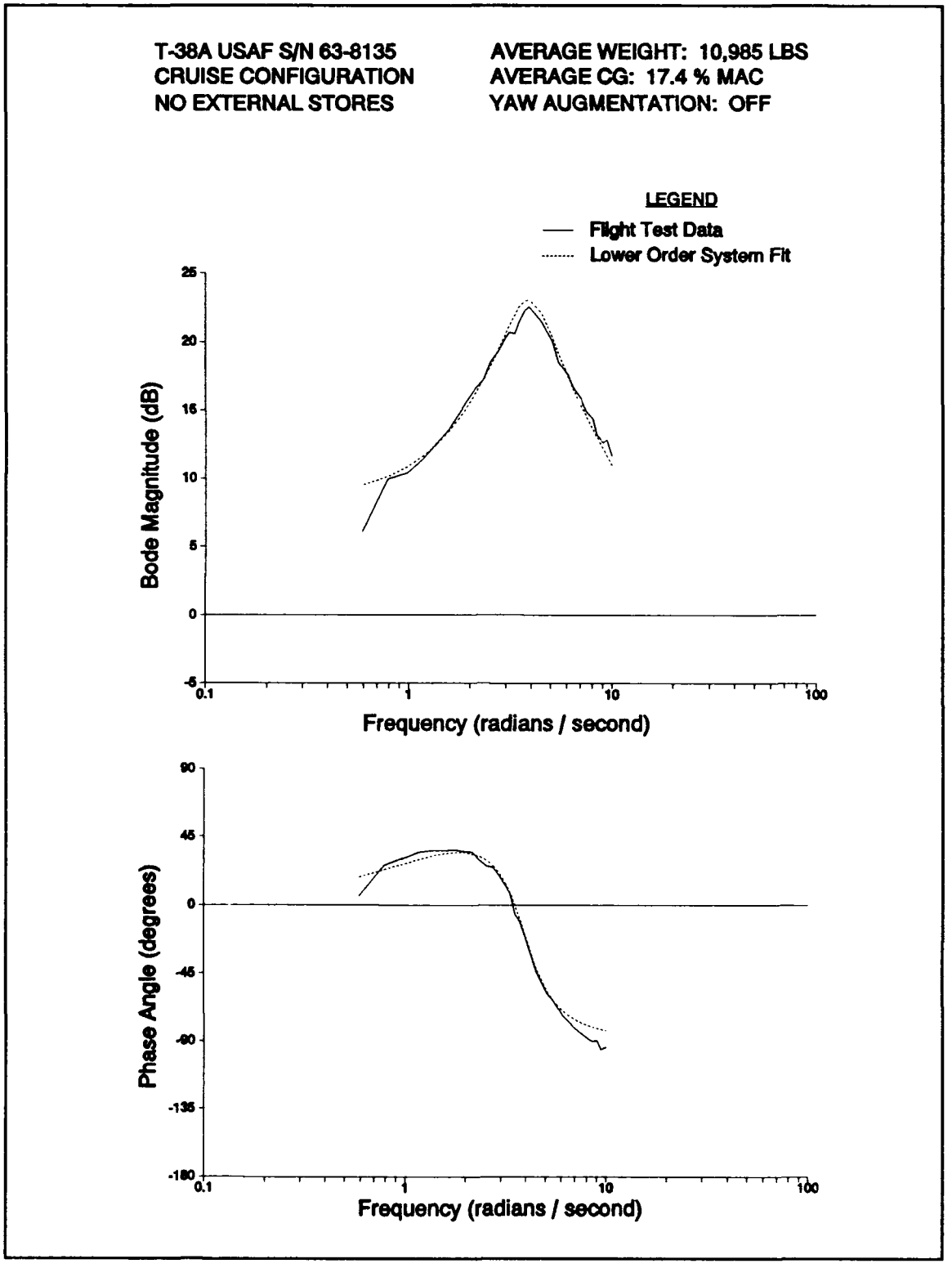

Figure 20. LOES Match, Test Point 4(a)

Finally, a comparison of equivalent second-order system parameters is presented in Tables 5 through 8 below for each test point. Classical analysis techniques were used to calculate damping ratio (ζ_{sp}) and natural frequency (ω_{sp}) from the classical pitch doublet inputs (1:8.48-8.53). The stability derivatives estimated by the PC and Cyber MMLE runs were used to approximate all four system parameters shown, as described in the last section. The LOES match discussed above was used to determine the system parameters from Cyber FRA.

In most cases, the PC results for each test point were less than five percent different from the Cyber. In all cases, the PC results were within 10 percent. Overall, the system parameters correlated extremely well among the different data collection methods and data reduction techniques.

Table 5. System Parameter Summary, Test Point 1(a)

PARAMETER	CLASSICAL	MMLE (PC)	MMLE (CYBER)	FRA/LOES (CYBER)
ζ_{sp}	0.31	0.3584	0.3475	0.3180
ω_{sp} (rad/s)	2.54	2.8599	2.8241	2.9680
K_θ	----	16.4185	15.9954	15.3100
$1/T_{\theta 2}$	----	1.1080	1.1852	1.2470

Table 6. System Parameter Summary, Test Point 2(a)

PARAMETER	CLASSICAL	MMLE (PC)	MMLE (CYBER)	FRA/LOES (CYBER)
ζ_{sp}	0.32	0.3412	0.3275	0.3103
ω_{sp} (rad/s)	3.26	4.0590	4.0193	3.9680
K_θ	----	29.4538	28.4696	25.6100
$1/T_{\theta 2}$	----	1.4363	1.5946	1.6680

Table 7. System Parameter Summary, Test Point 3(a)

PARAMETER	CLASSICAL	MMLE (PC)	MMLE (CYBER)	FRA/LOES (CYBER)
ζ_{sp}	0.31	0.2872	0.2703	0.3196
ω_{sp} (rad/s)	2.64	2.8034	2.7365	2.7820
K_θ	----	18.4411	17.8397	18.4400
$1/T_{\theta 2}$	----	0.8924	0.8505	0.7041

Table 8. System Parameter Summary, Test Point 4(a)

PARAMETER	CLASSICAL	MMLE (PC)	MMLE (CYBER)	FRA/LOES (CYBER)
ζ_{sp}	0.29	0.3079	0.3244	0.2880
ω_{sp} (rad/s)	3.28	3.8058	3.8660	3.9520
K_θ	----	32.0179	33.4768	30.6200
$1/T_{\theta 2}$	----	1.2477	1.1767	1.3750

VI. Conclusions and Recommendations

A system of macro files has been developed for performing Modified Maximum Likelihood Estimation of aircraft stability derivatives using a personal computer. The process is completed in the environment of the PC Matlab personal computer program, with integration of flight test data files from the Test Pilot School data reduction facility.

Results from this personal computer based approach have been compared with results from the mainframe computer program (MMLE3) which is considered the standard for parameter estimation at the Air Force Flight Test Center. The personal computer results were generally well within the uncertainty levels of the mainframe estimates. A comparison of second-order system parameters obtained from classical analysis, stability derivatives extracted using both MMLE programs, and frequency response analysis techniques showed that the system parameters correlated extremely well among the different data collection methods and data reduction techniques. Overall, the accuracy of the personal computer method was satisfactory, verifying the feasibility of the personal computer based approach to aircraft parameter identification.

The quality of flight test inputs was critical to successful parameter estimation. Frequency sweeps were particularly challenging to perform. Practice helped the quality of the inputs significantly, and pilots felt that the time spent in the ground simulator prior to flight was invaluable. Ground simulator training should be accomplished before collecting inflight data using parameter estimation flight test techniques.

The best data available for moments of inertia were in a Northrop Corporation report from 1958. Future attempts to perform parameter estimation on the T-38A should calculate data which more closely represents the flight test aircraft. This would ensure estimates are as close as possible to those of the actual aircraft.

Other possibilities for future research projects include investigation of including boom bending and upwash corrections to the angle of attack measurements. This would improve the time history match of the angle of attack trace. Another possible area for improvement is correction for non-zero lateral and directional parameters. The current system assumes that for a longitudinal pulse there are no lateral or directional responses. It would be an improvement to the flexibility of the model to incorporate these corrections into the model. Finally, PC MMLE capability for the lateral-directional axes could be investigated.

Appendix A: Flight Test Information

Table A1. Test Parameter Ranges and Resolutions

PARAMETER	RANGE	RESOLUTION
TIME	----	millisecond
ELEVATOR POSITION	-8 to +16 deg	0.03 deg
LEFT AILERON POSITION	-35 to +25 deg	0.06 deg
RIGHT AILERON POSITION	-25 to +35 deg	0.06 deg
RUDDER POSITION	±30 deg	0.06 deg
LONG. STICK POSITION	-5 to +7 in	0.02 in
LATERAL STICK POSITION	±8 in	0.02 in
RUDDER PEDAL POSITION	±5 in	0.01 in
LONG. STICK FORCE	±70 lb	0.20 lb
LATERAL STICK FORCE	±35 lb	0.10 lb
RUDDER PEDAL FORCE	±150 lb	0.20 lb
ANGLE OF ATTACK	-10 to +40 deg	0.05 deg
ANGLE OF SIDESLIP	±10 deg	0.02 deg
PITCH ANGLE	±60 deg	0.20 deg
BANK ANGLE	±90 deg	0.40 deg
PITCH RATE	±50 deg/s	0.10 deg/s
ROLL RATE	±100 deg/s	0.20 deg/s
YAW RATE	±50 deg/s	0.10 deg/s
NORMAL ACCELERATION	-2 to +5 g	0.01 g
LONG. ACCELERATION	-1 to +1 g	0.01 g
LATERAL ACCELERATION	-1 to +1 g	0.01 g

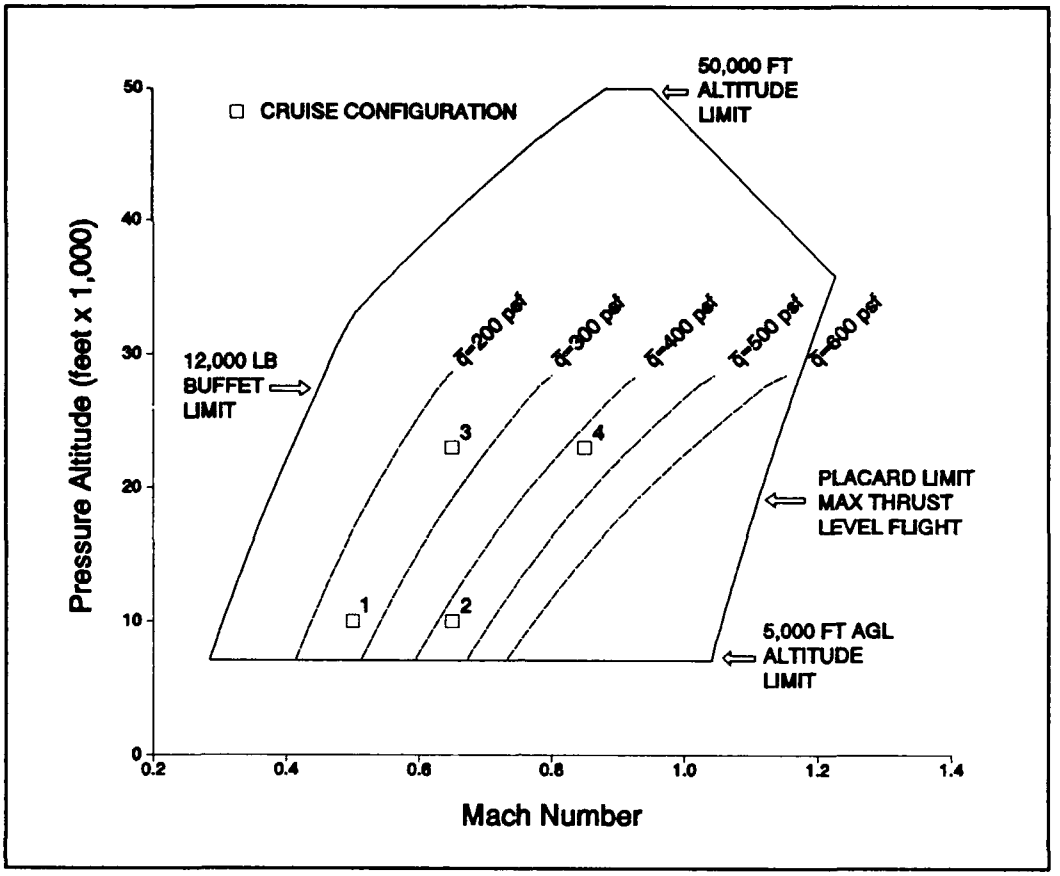

Figure A1. Flight Test Envelope

Table A2. Test Point Description

TEST POINT	PRESSURE ALTITUDE	MACH NUMBER	DYNAMIC PRESSURE
1 (a), (b), and (c)	10,000 ft	0.50	250 psf
2 (a), (b), and (c)	10,000 ft	0.65	430 psf
3 (a), (b), and (c)	23,000 ft	0.65	250 psf
4 (a), (b), and (c)	23,000 ft	0.85	430 psf

Note: At each test point, the (a), (b), and (c) correspond to angles of attack of 1g trim, trim + 2 degrees, and trim + 4 degrees, respectively.

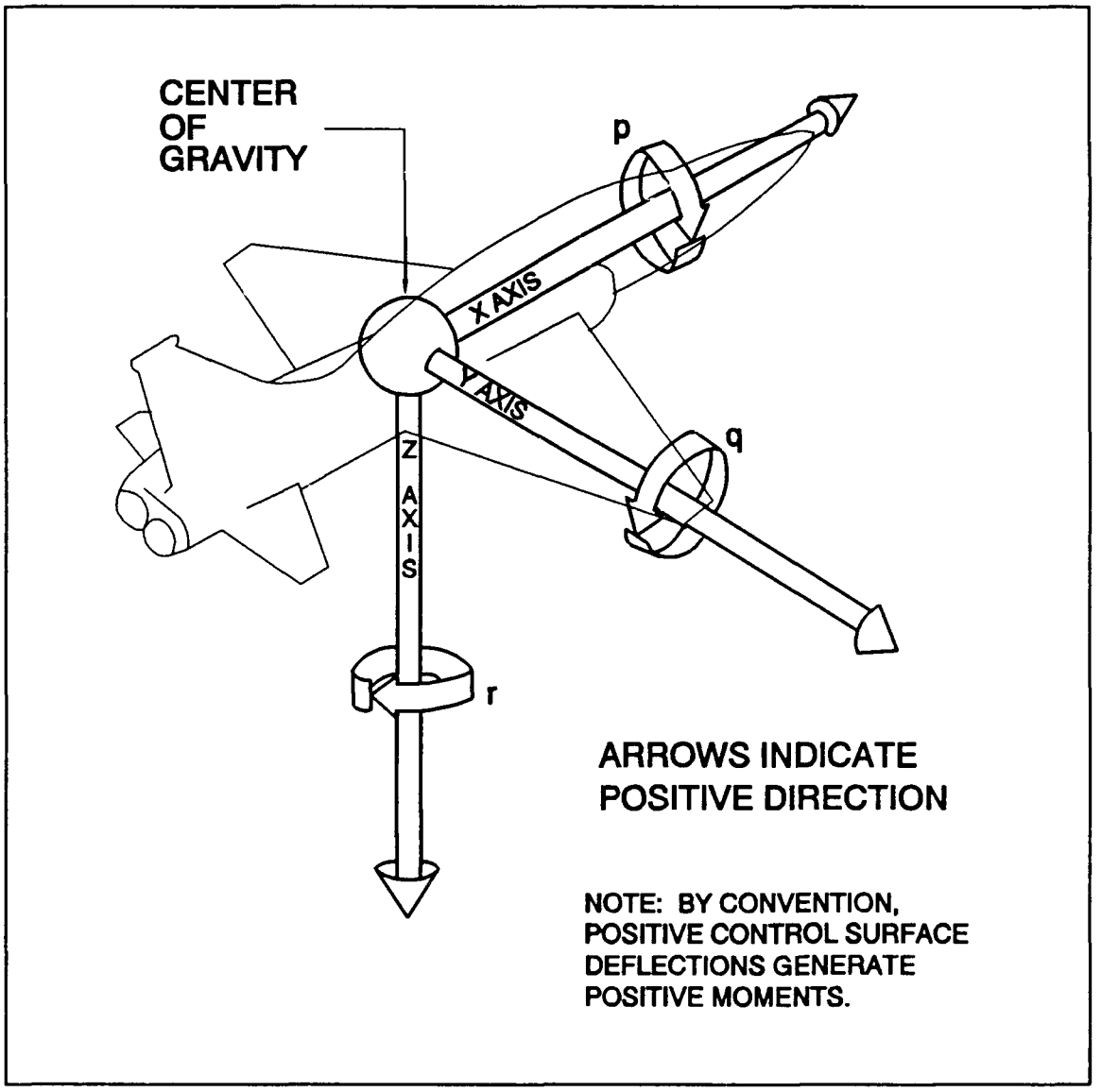

Figure A2. Test Pilot School Sign Convention

Appendix B. Supplemental Flight Test Data

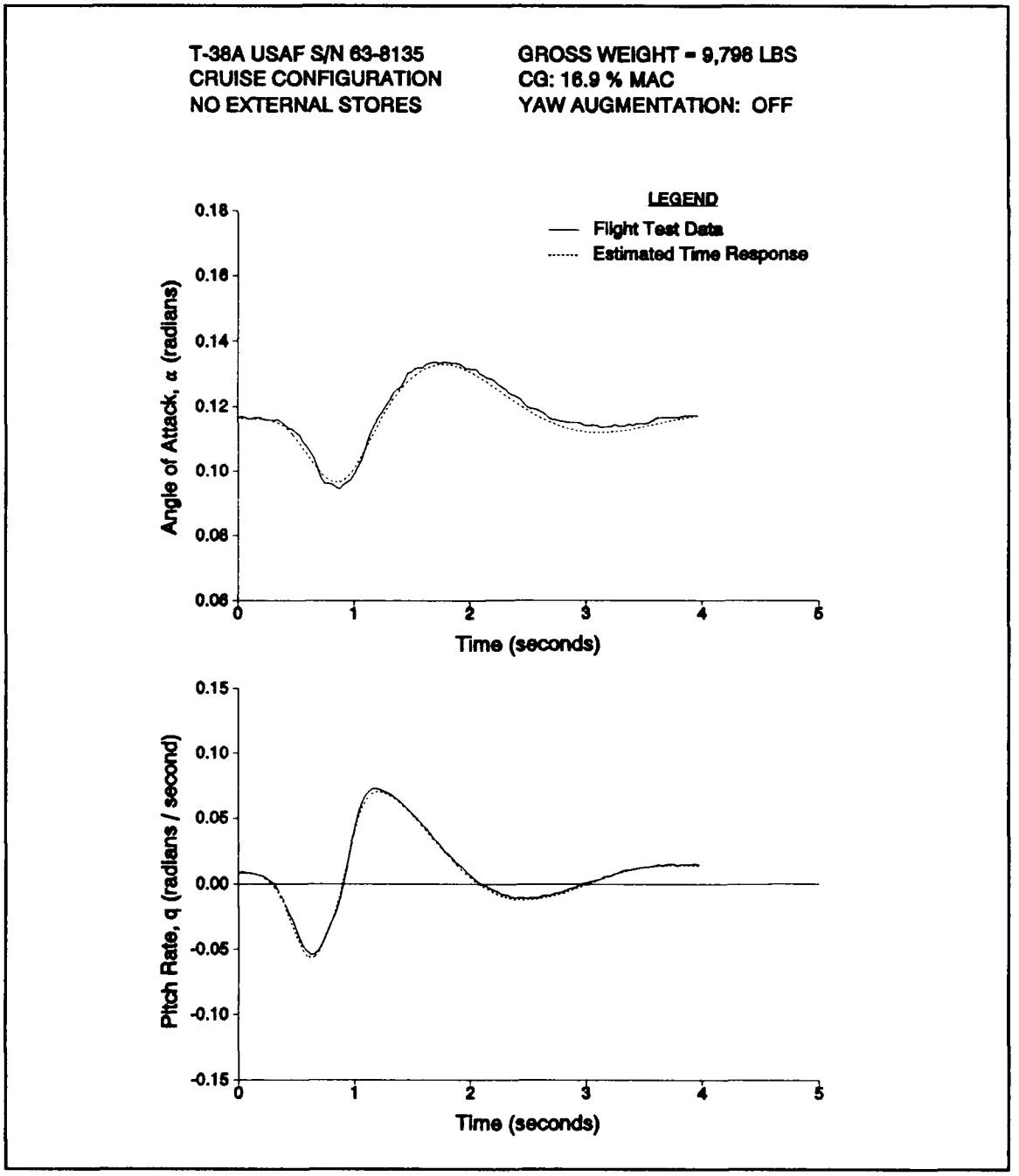

Figure B1. Time History Match, Test Point 1(b) (Sheet 1 of 2)

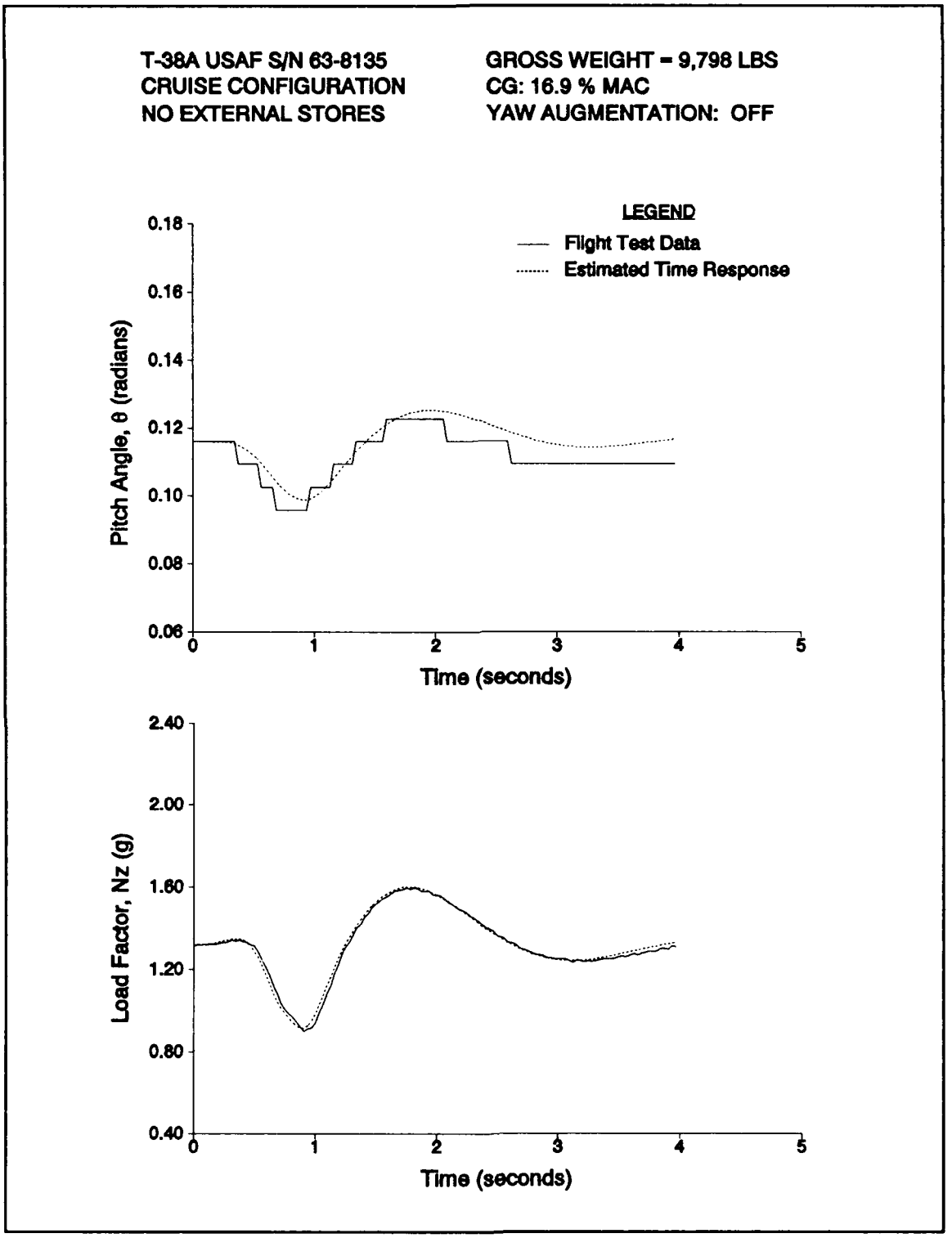

Figure B1. Time History Match, Test Point 1(b) (Sheet 2 of 2)

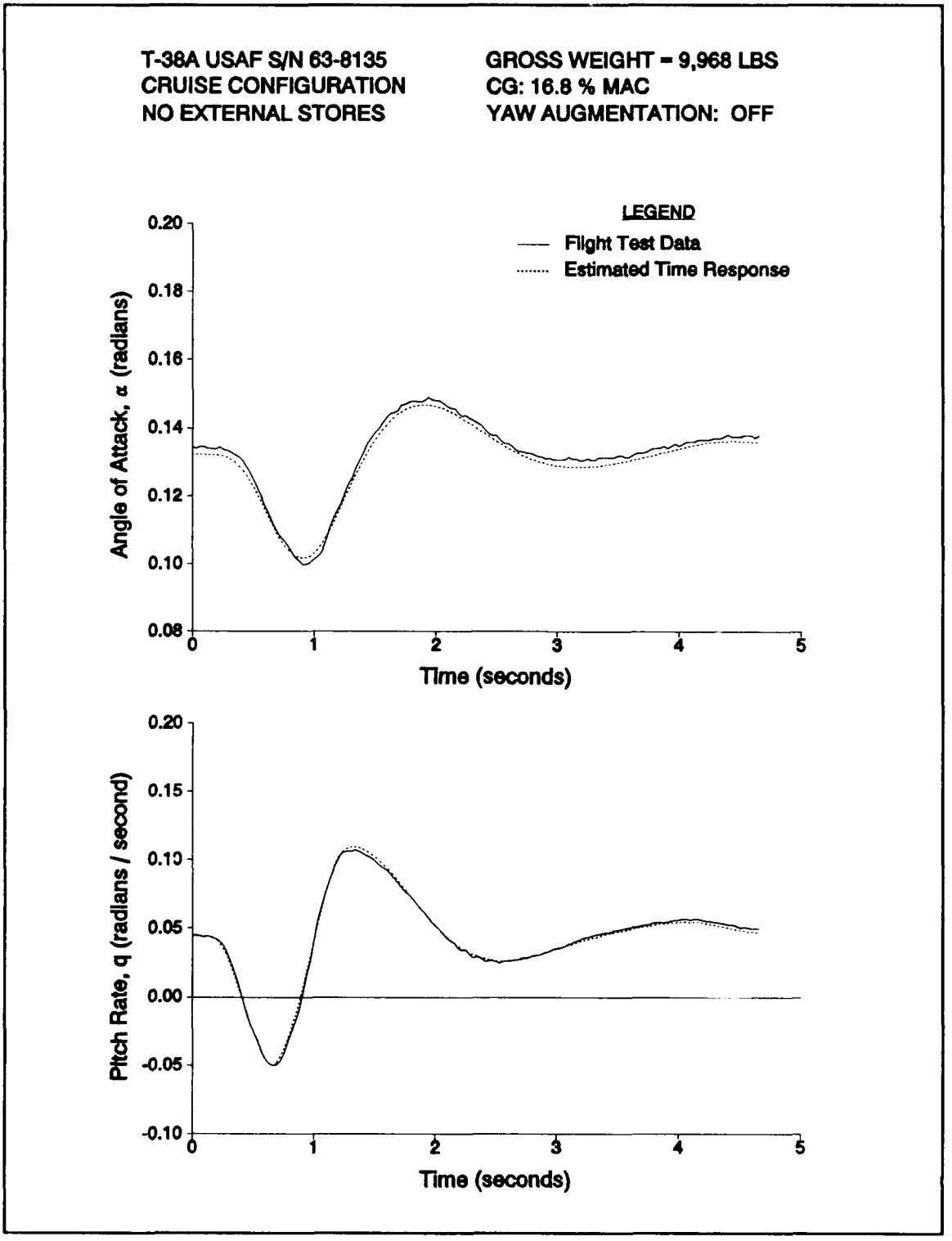

Figure B2. Time History Match, Test Point 1(c) (Sheet 1 of 2)

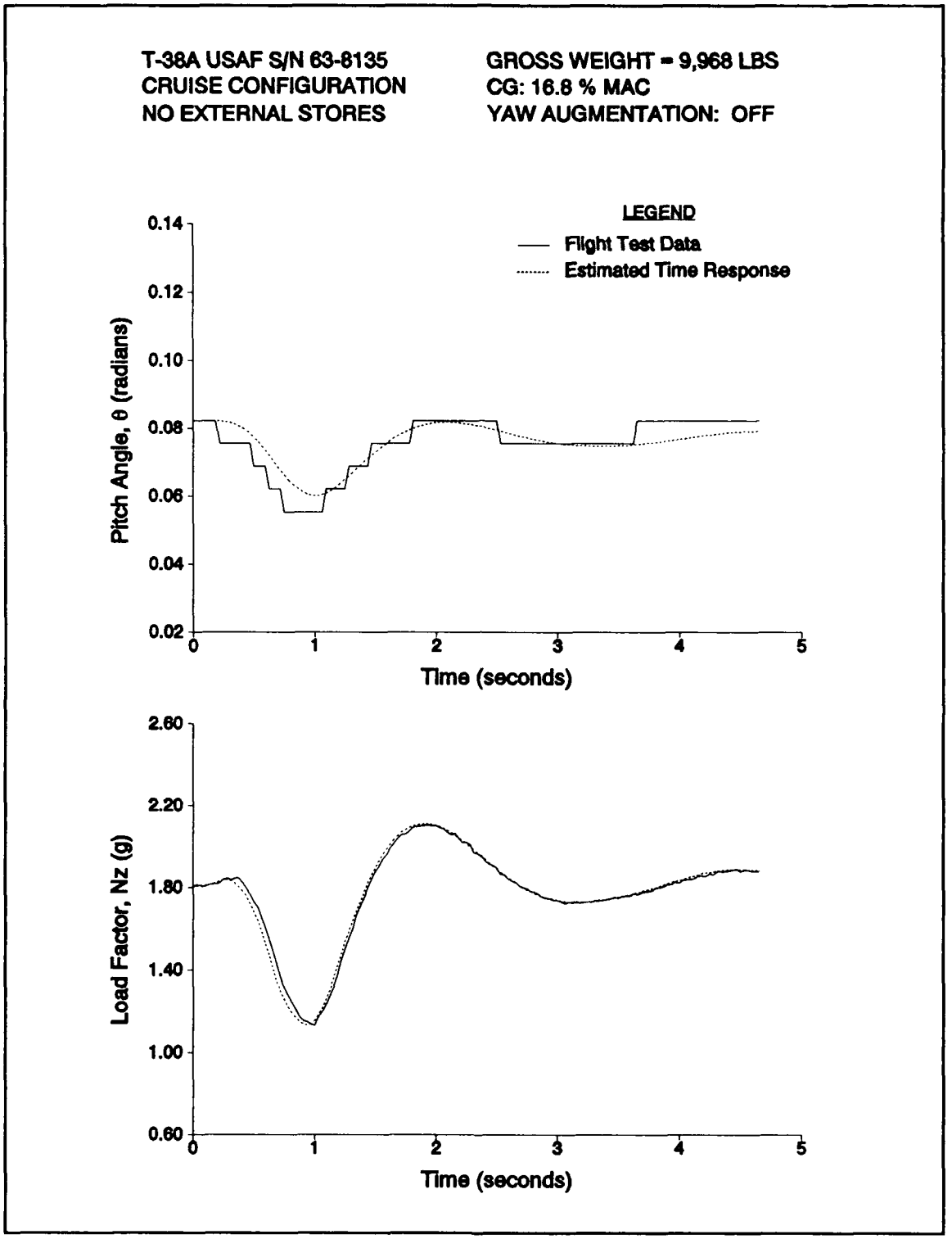

Figure B2. Time History Match, Test Point 1(c) (Sheet 2 of 2)

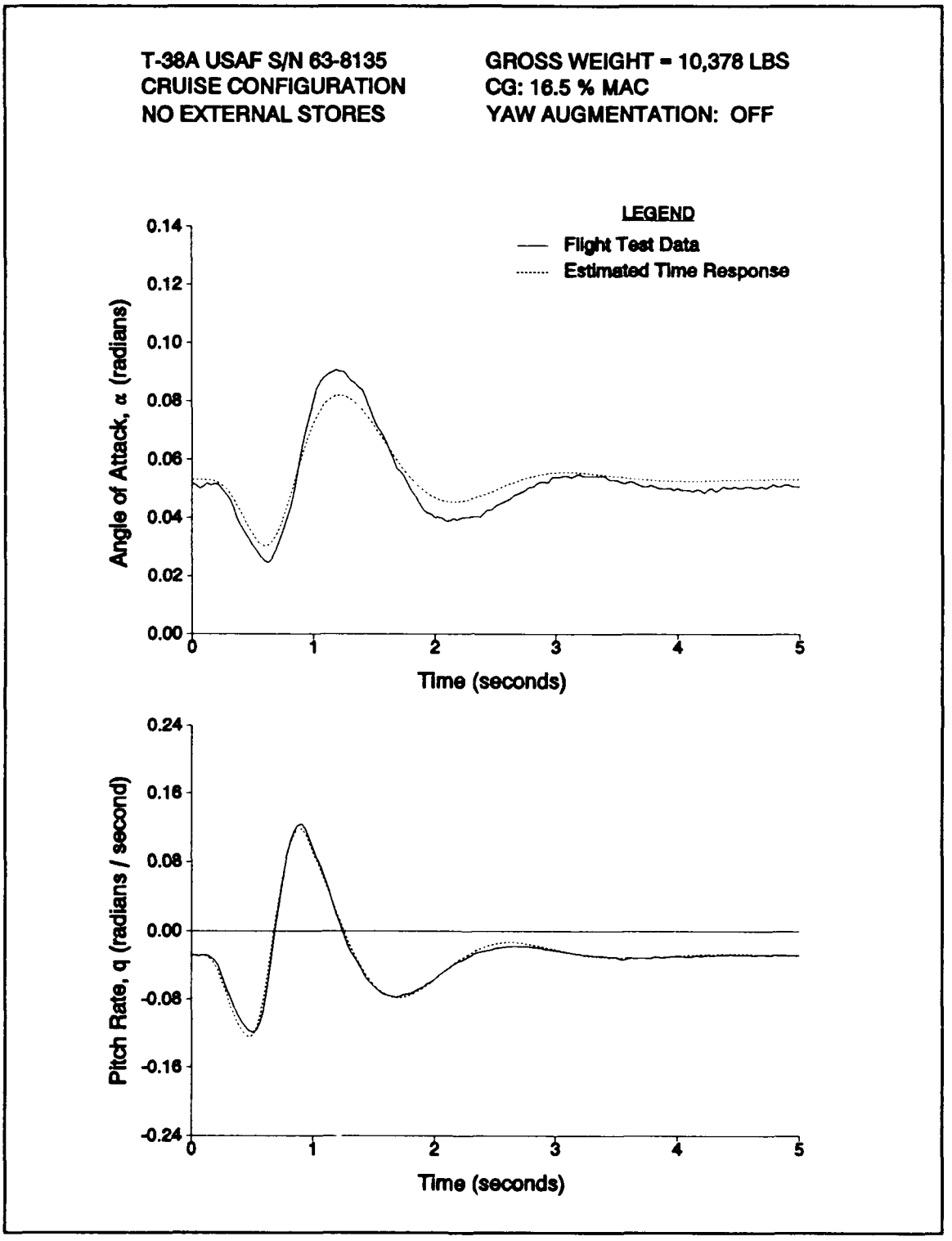

Figure B3. Time History Match, Test Point 2(a) (Sheet 1 of 2)

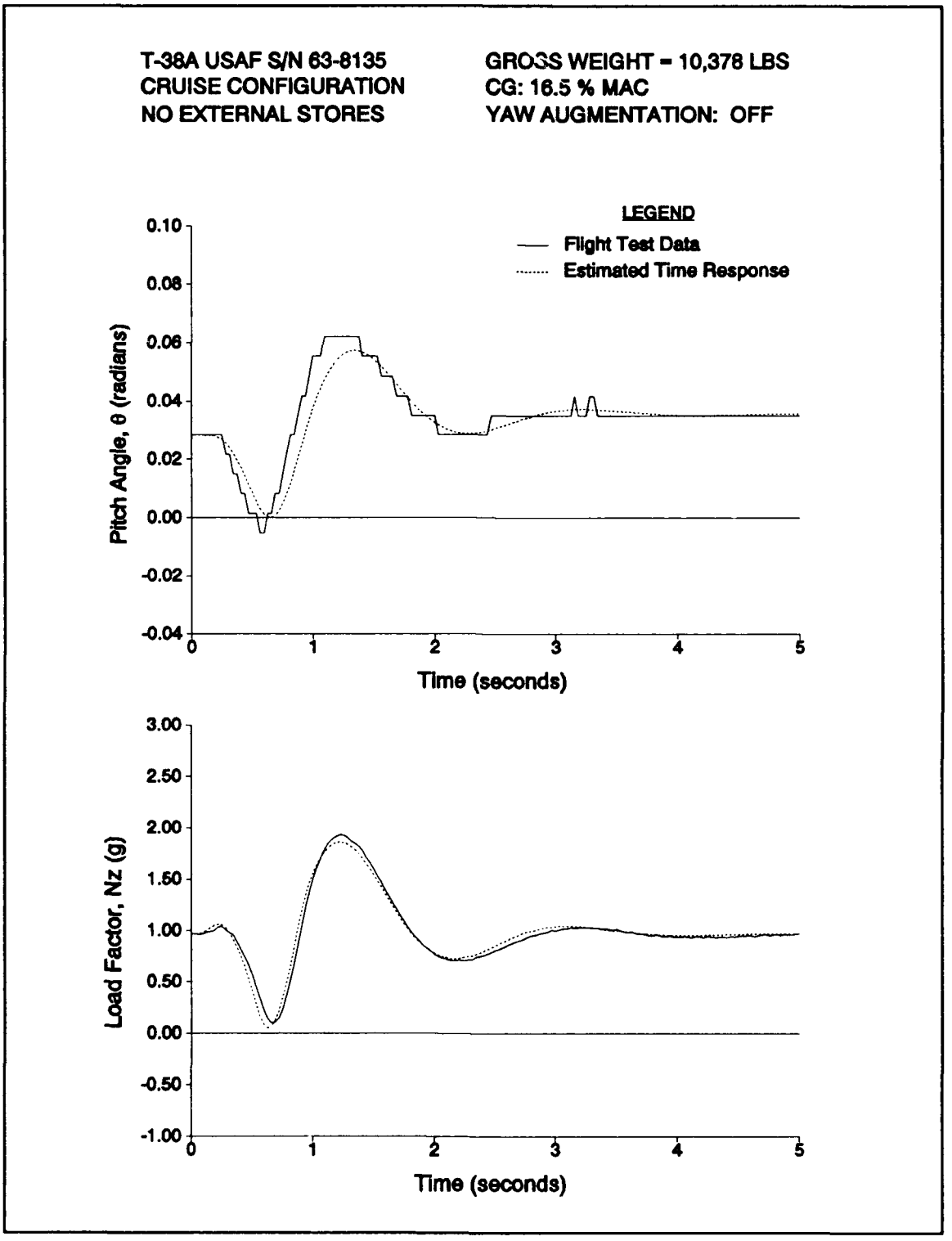

Figure B3. Time History Match, Test Point 2(a) (Sheet 2 of 2)

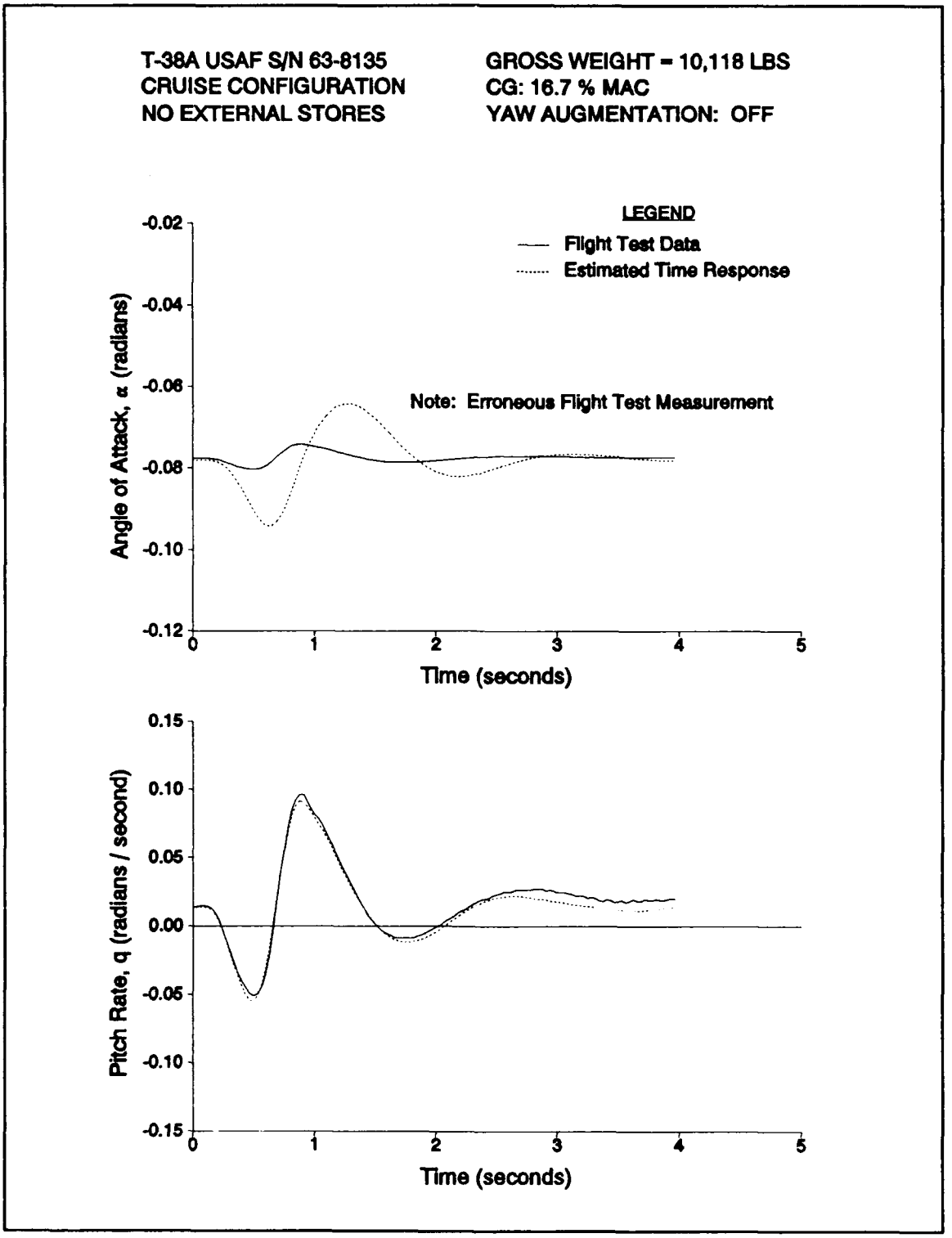

Figure B4. Time History Match, Test Point 2(b) (Sheet 1 of 2)

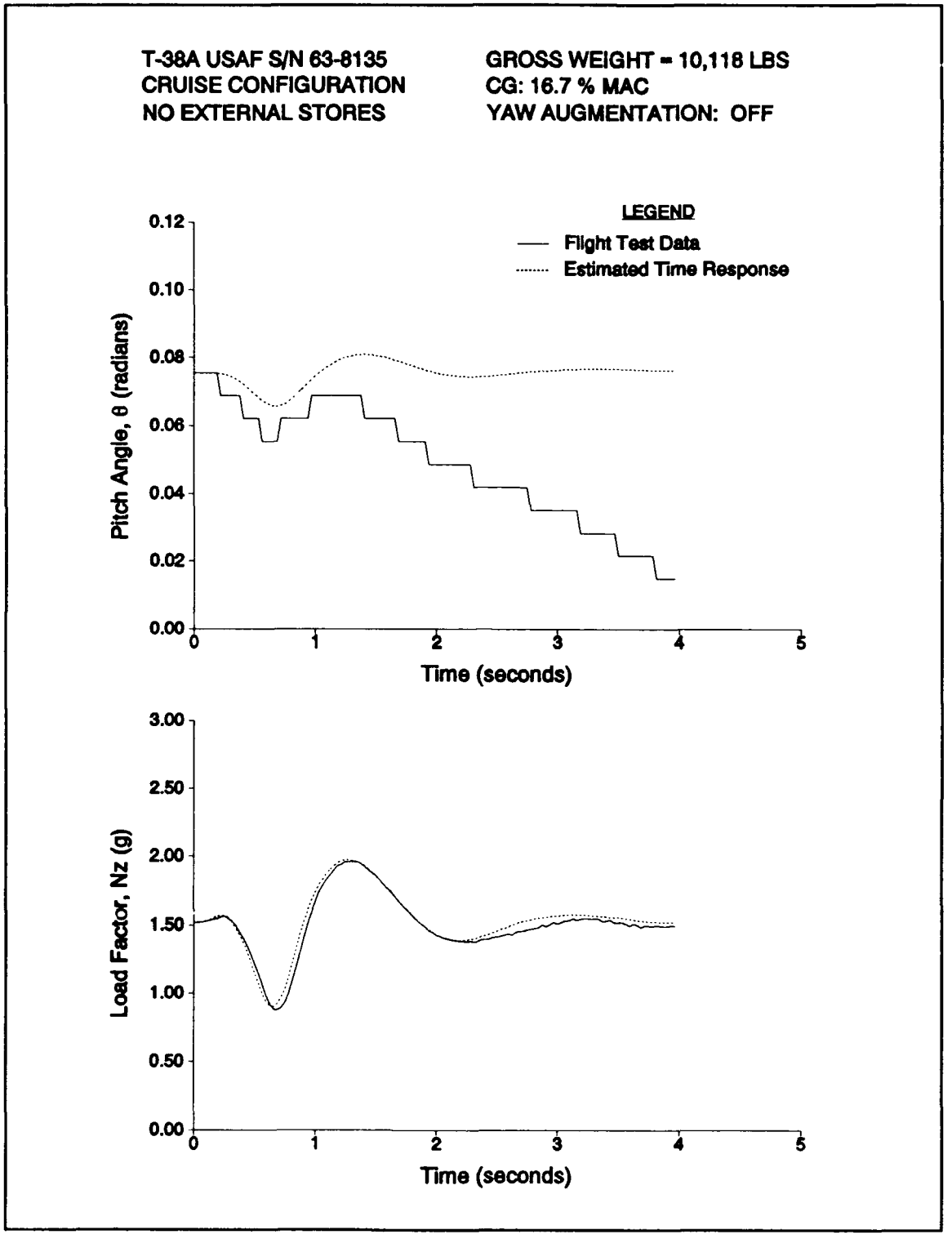

Figure B4. Time History Match, Test Point 2(b) (Sheet 2 of 2)

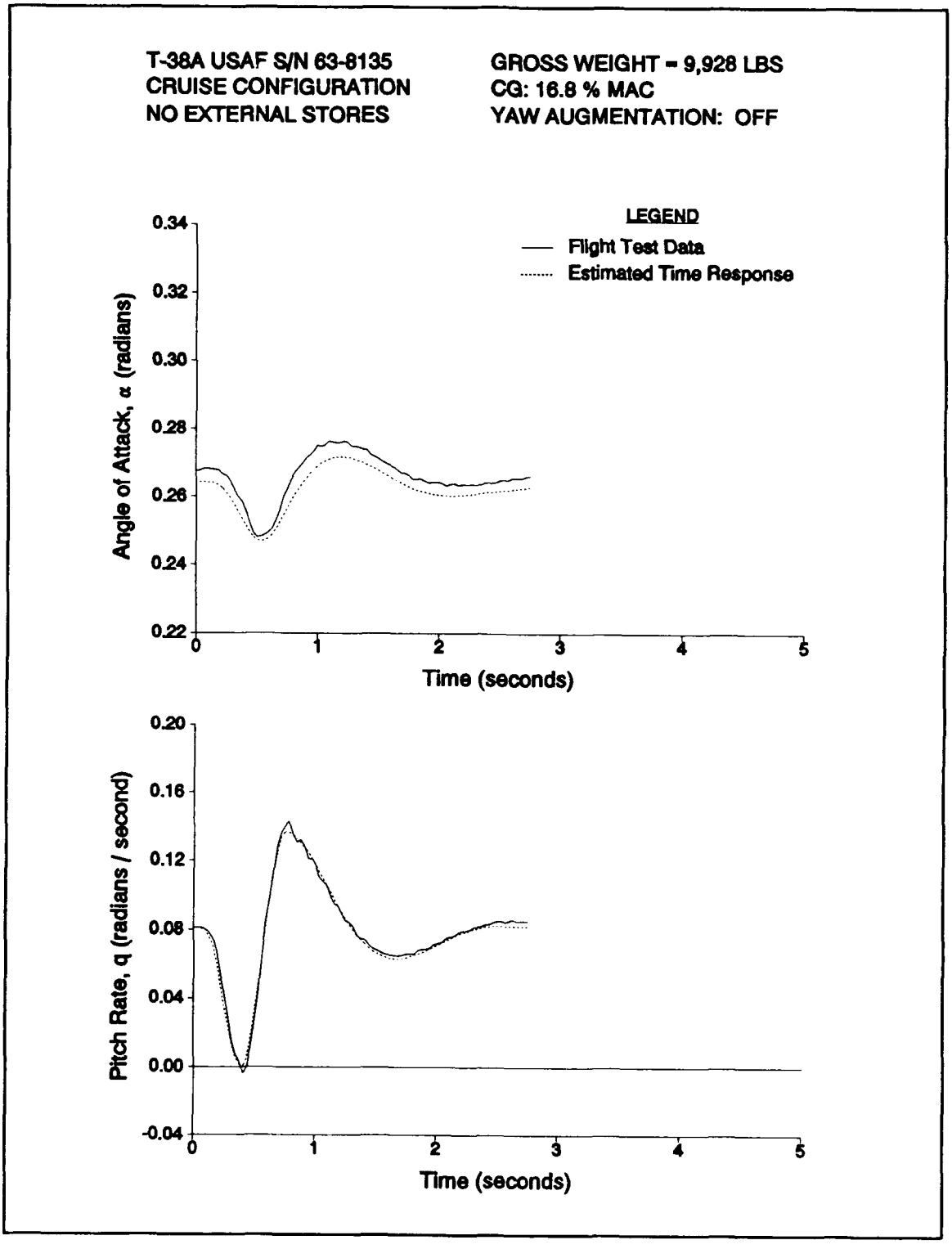

Figure B5. Time History Match, Test Point 2(c) (Sheet 1 of 2)

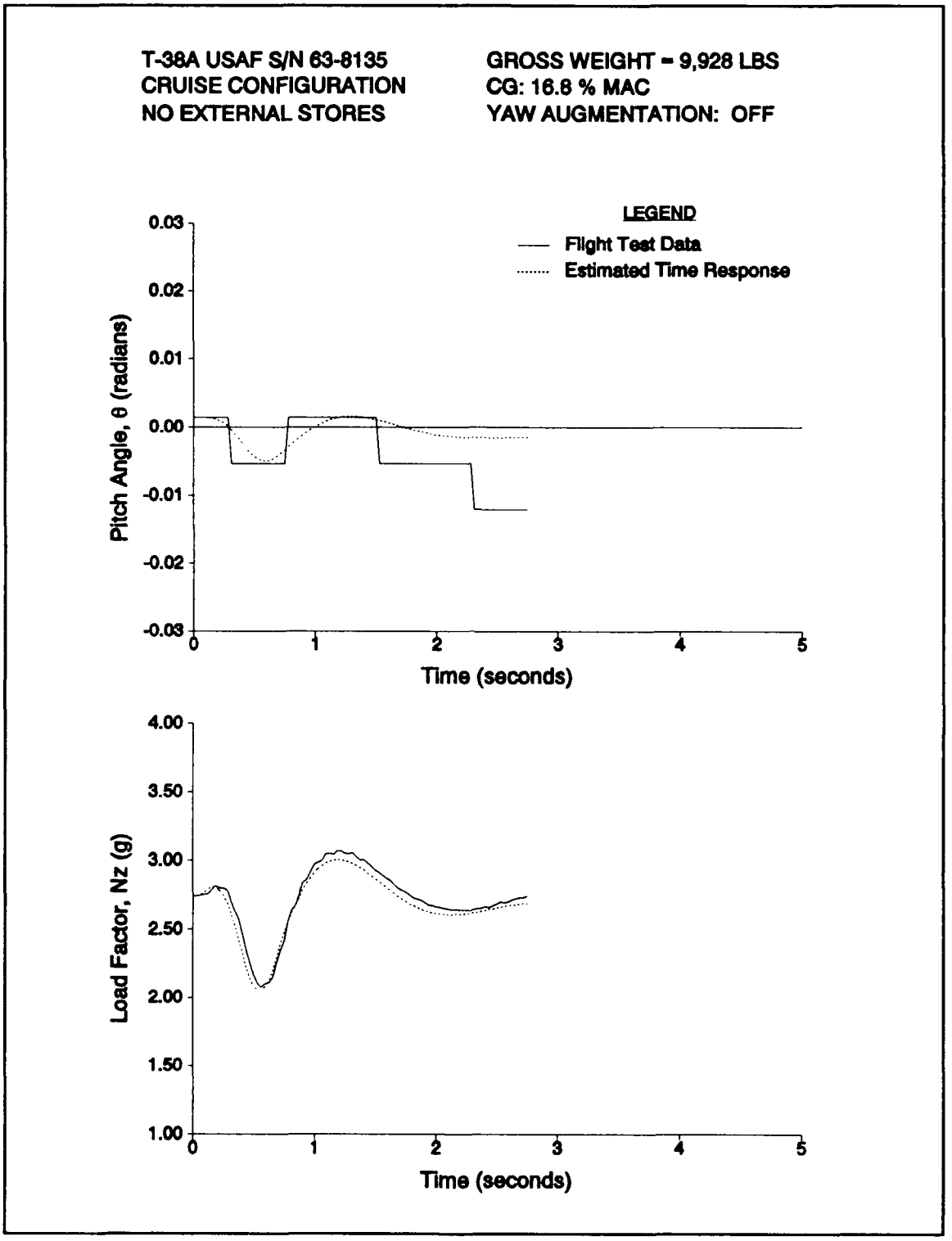

Figure B5. Time History Match, Test Point 2(c) (Sheet 2 of 2)

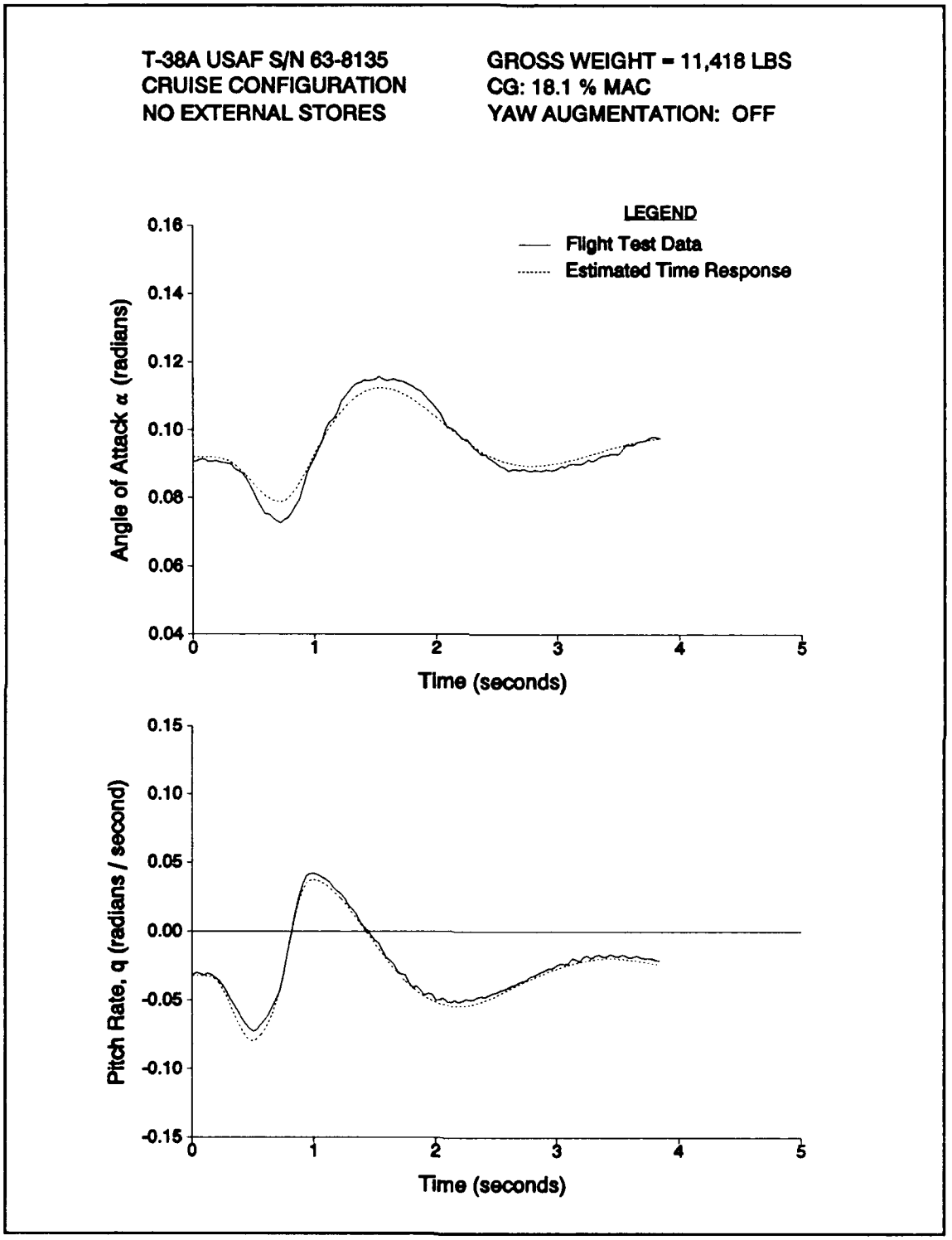

Figure B6. Time History Match, Test Point 3(a) (Sheet 1 of 2)

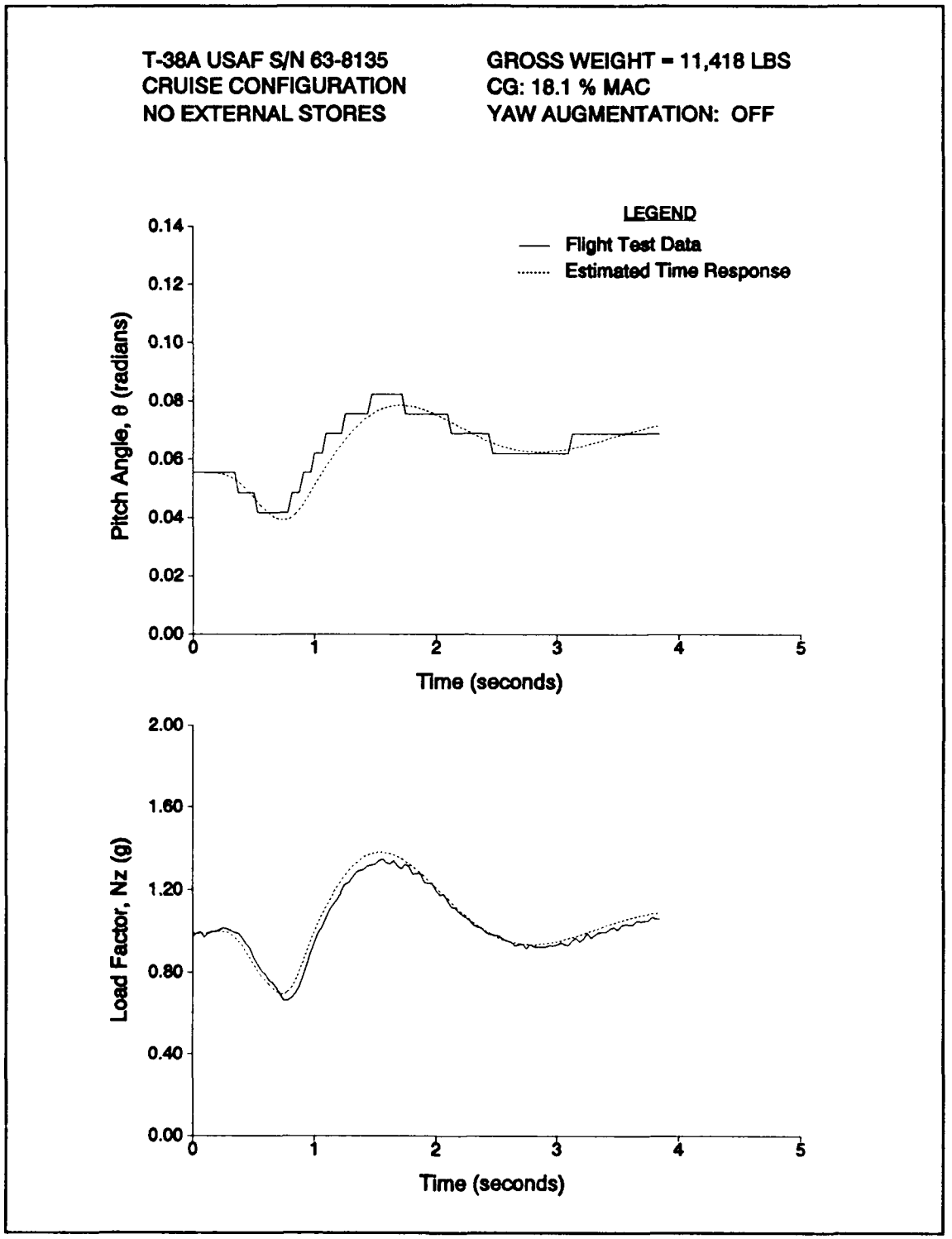

Figure B6. Time History Match, Test Point 3(a)
(Sheet 2 of 2)

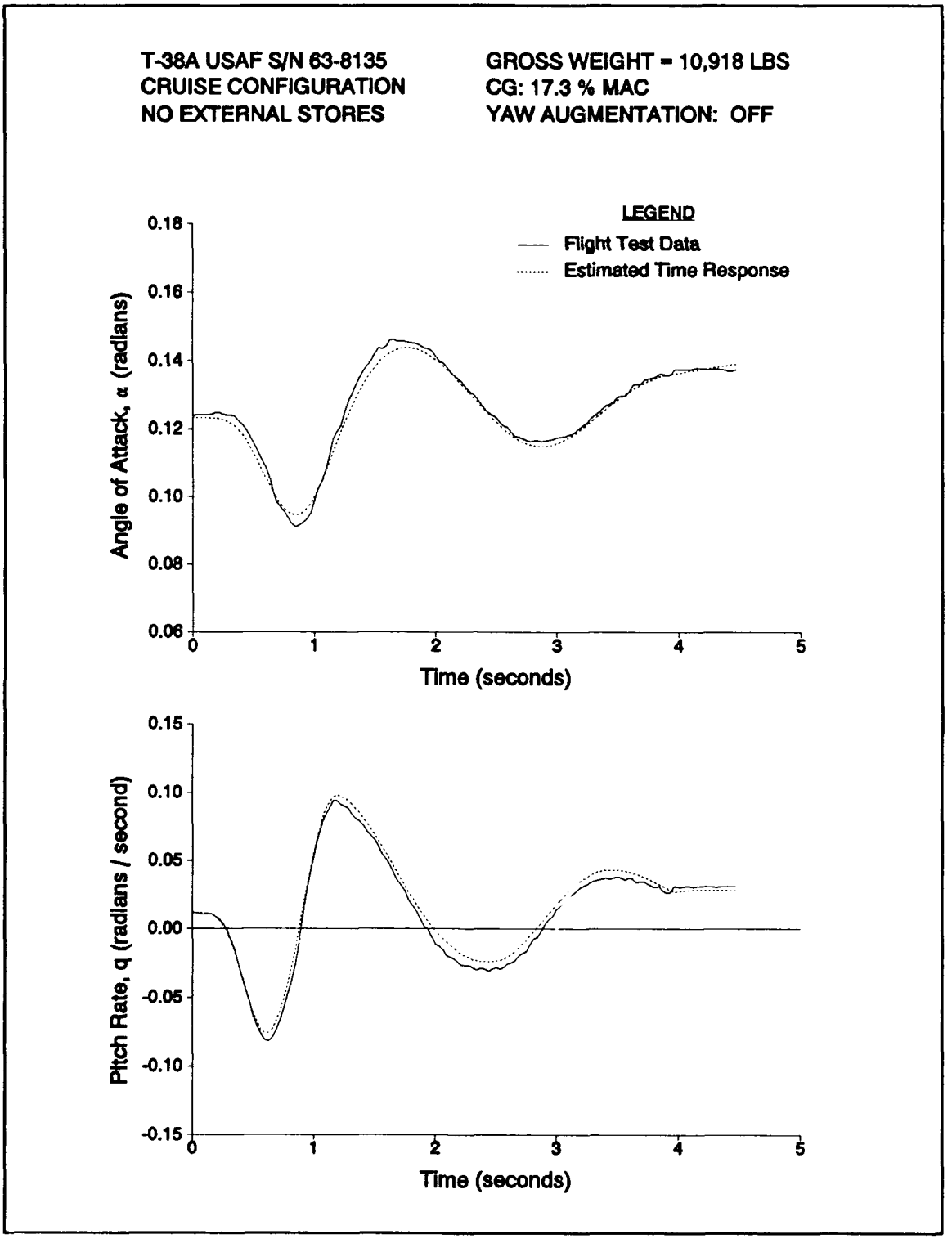

Figure B7. Time History Match, Test Point 3(b) (Sheet 1 of 2)

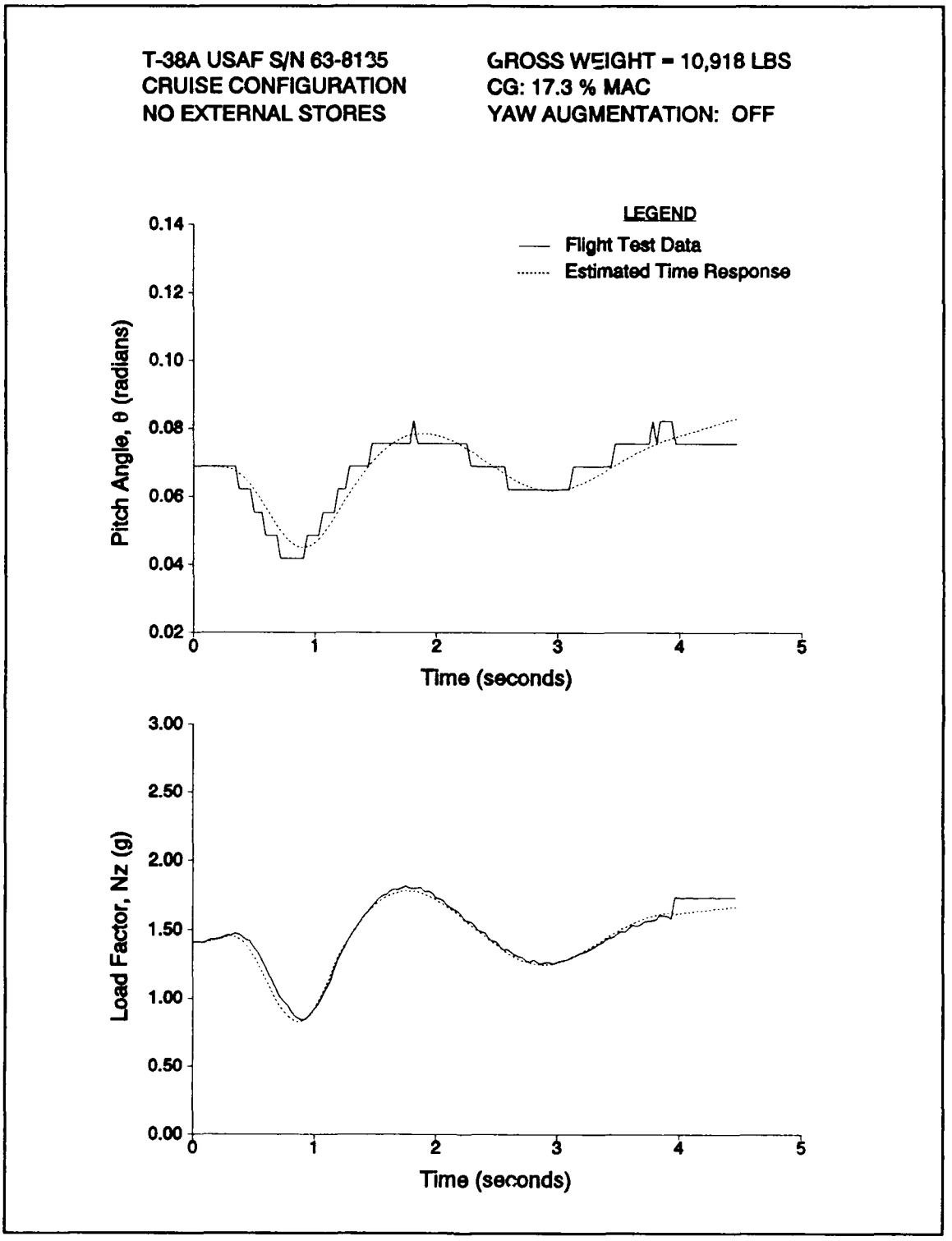

Figure B7. Time History Match, Test Point 3(b) (Sheet 2 of 2)

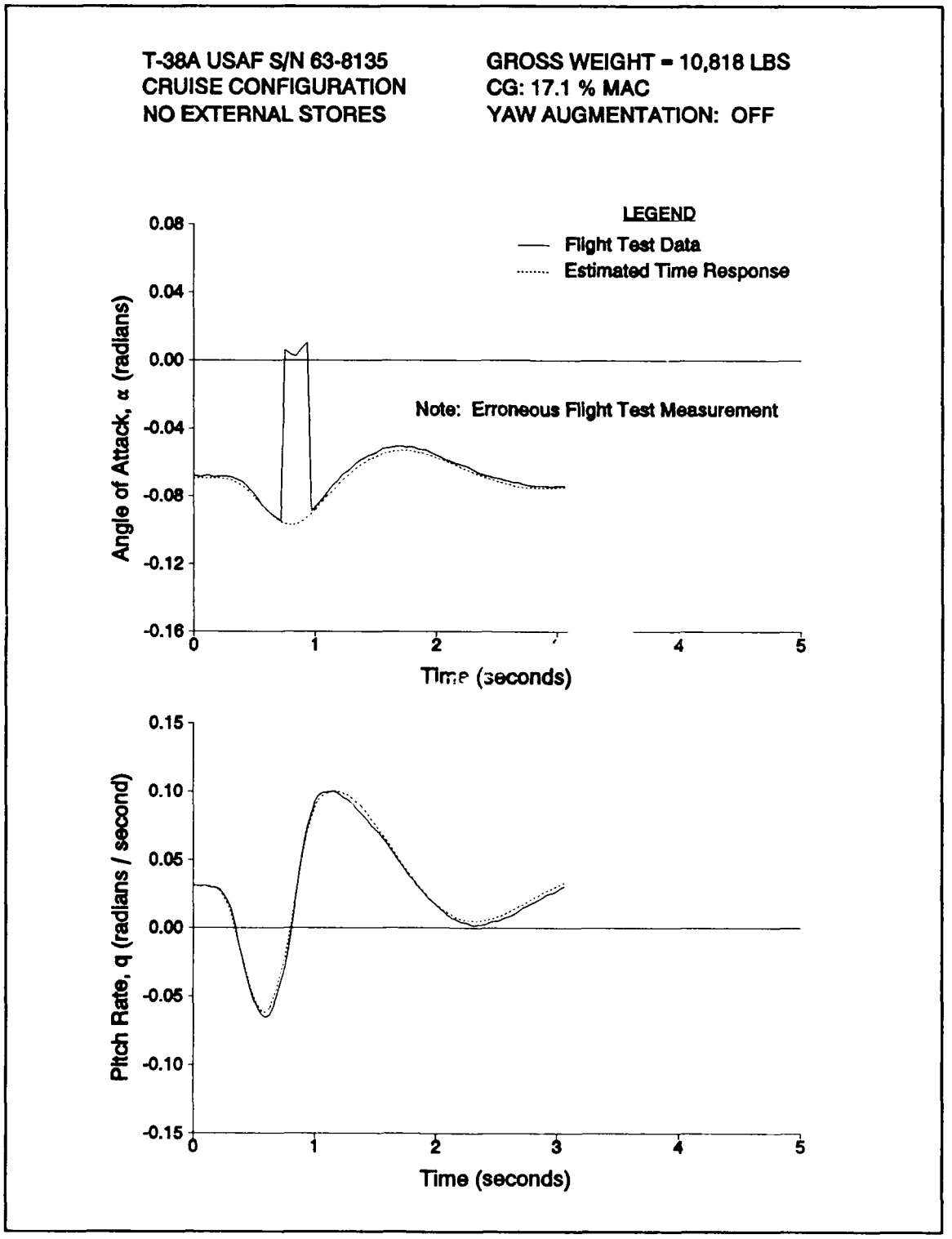

Figure B8. Time History Match, Test Point 3(c) (Sheet 1 of 2)

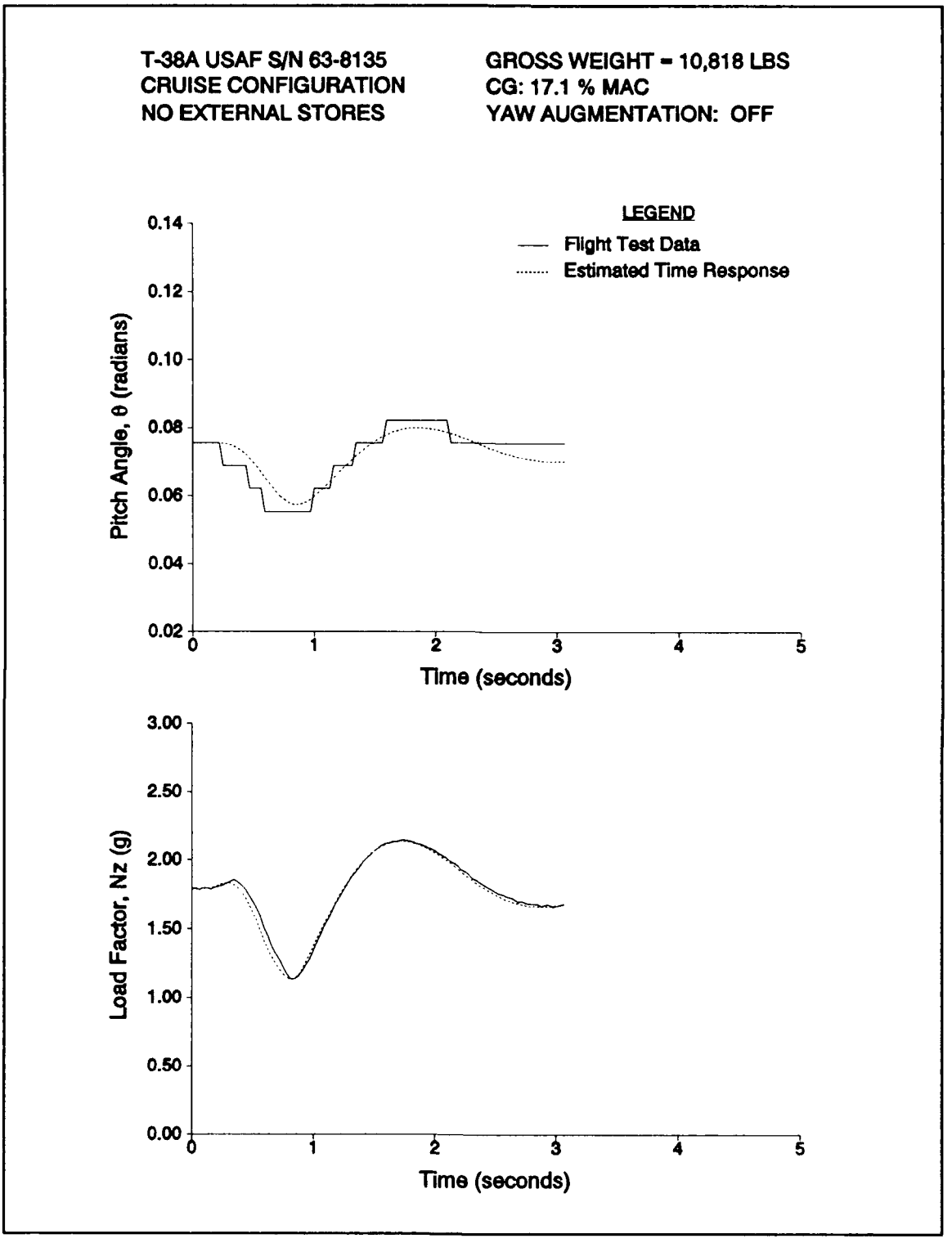

Figure B8. Time History Match, Test Point 3(c) (Sheet 2 of 2)

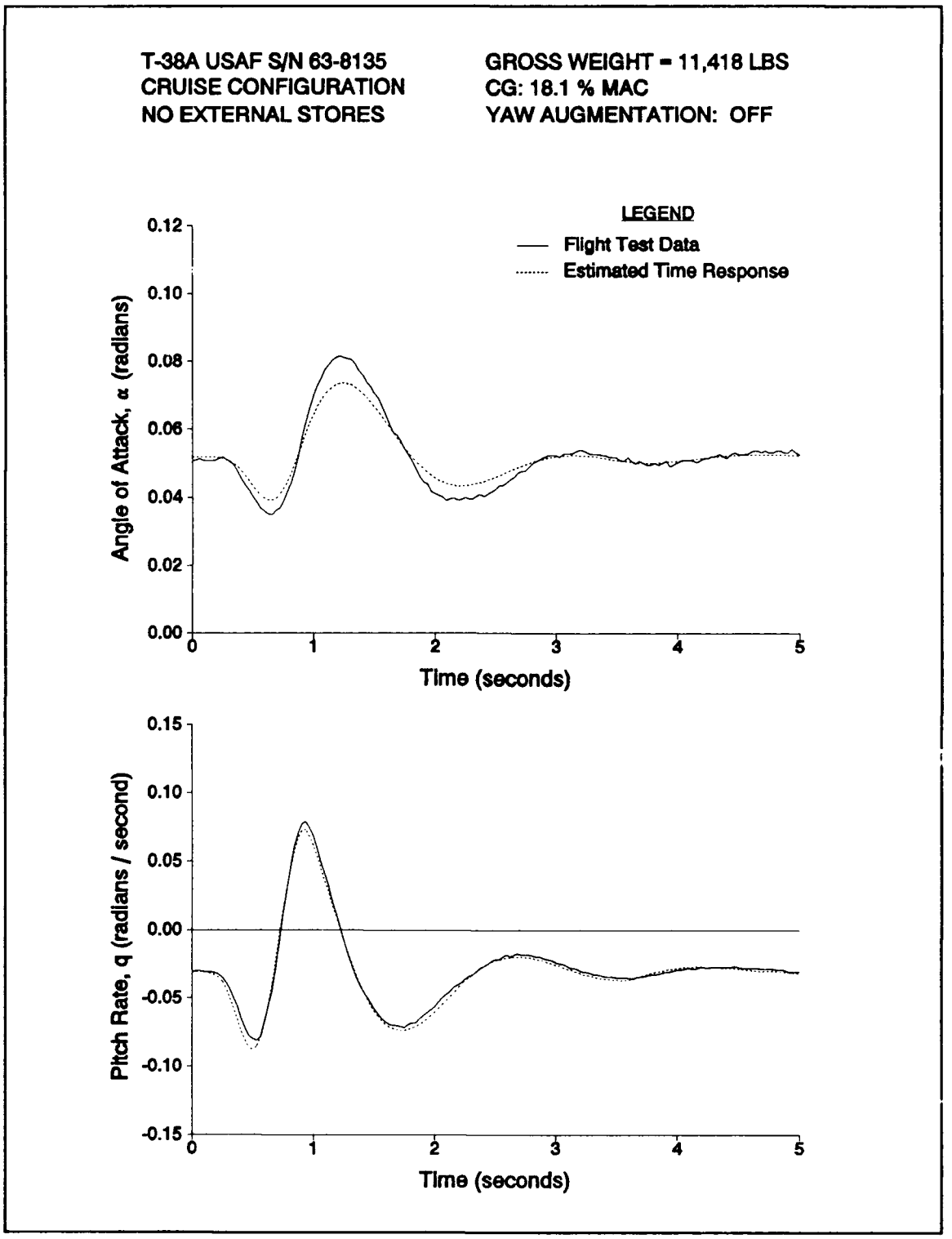

Figure B9. Time History Match, Test Point 4(a) (Sheet 1 of 2)

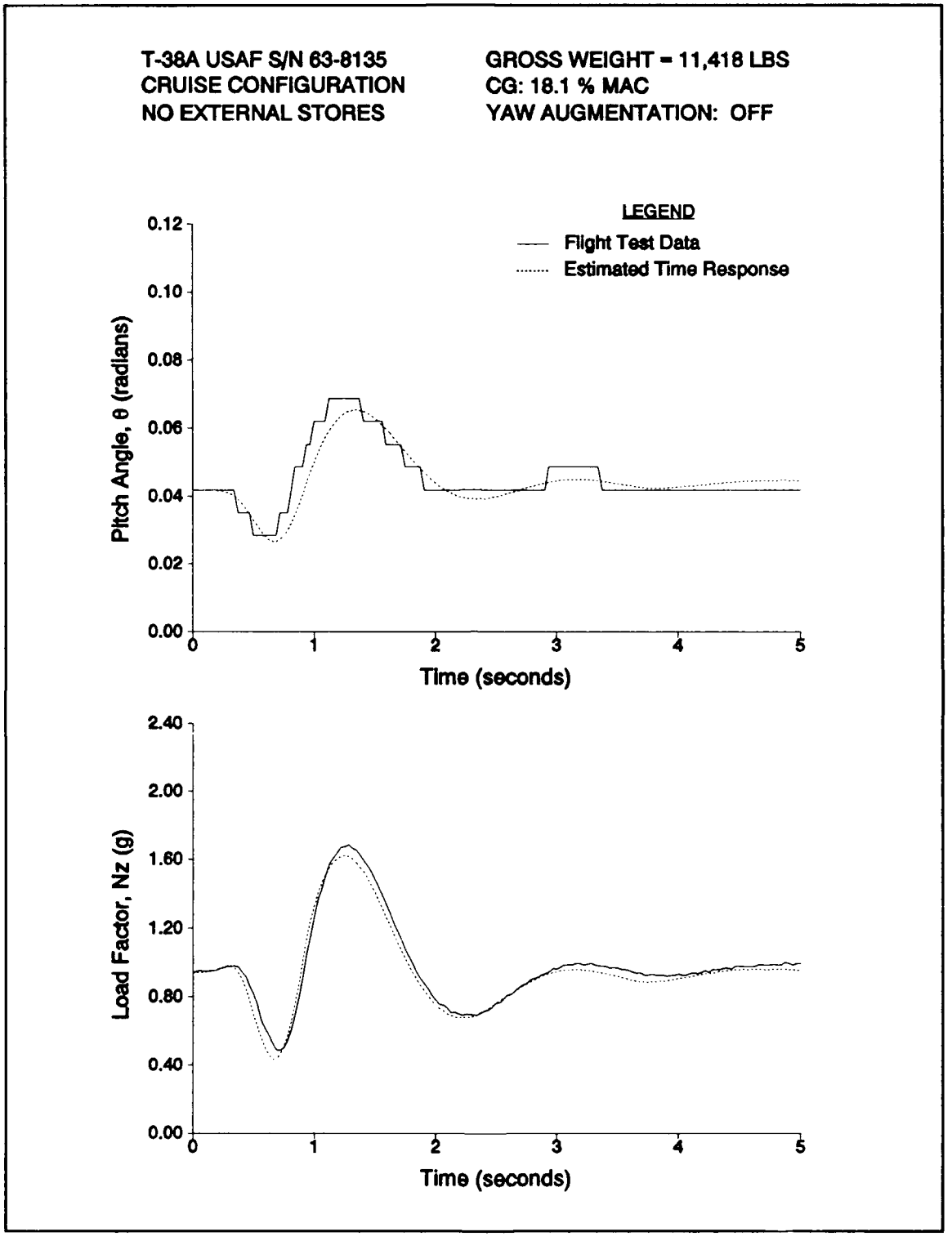

Figure B9. Time History Match, Test Point 4(a) (Sheet 2 of 2)

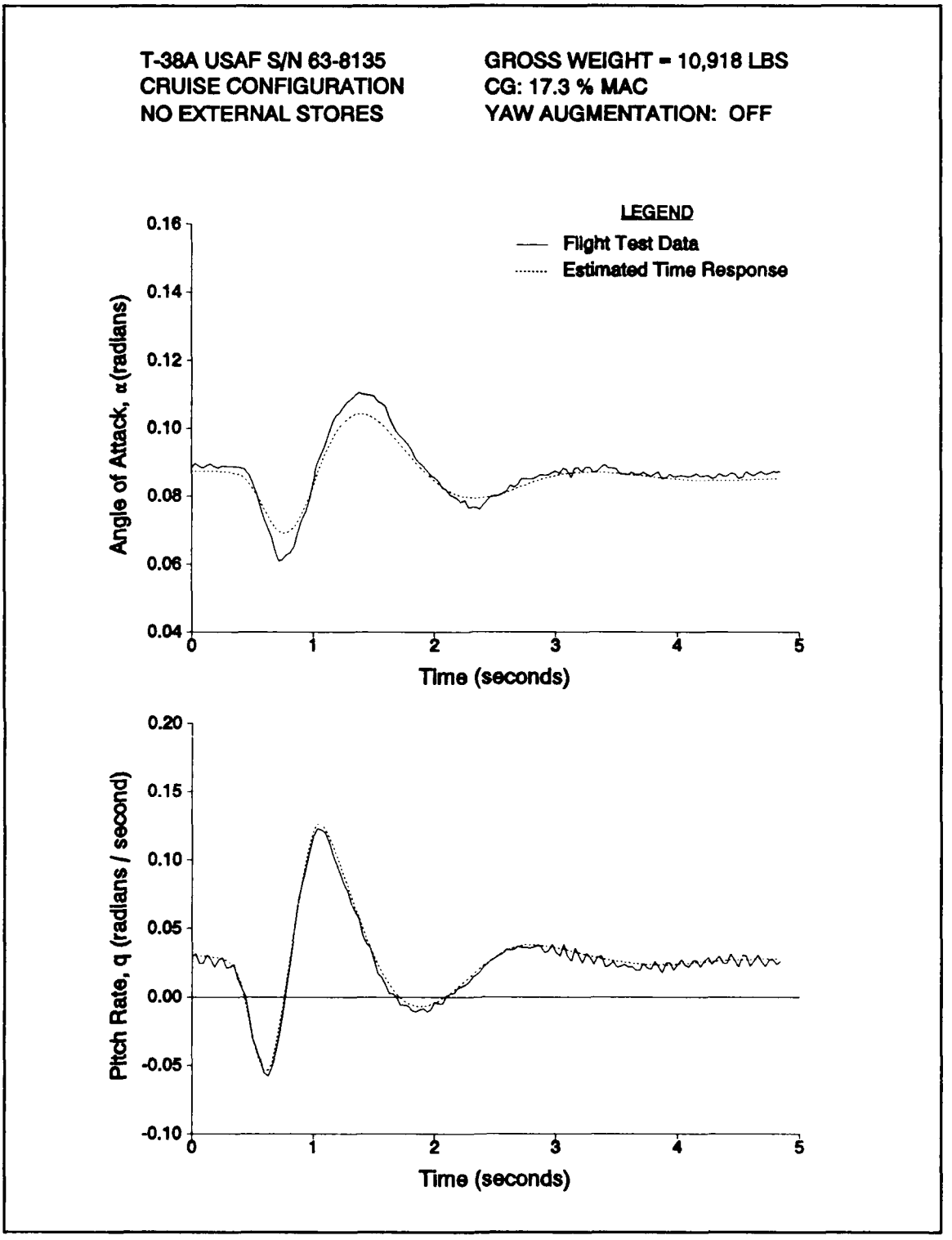

Figure B10. Time History Match, Test Point 4(b) (Sheet 1 of 2)

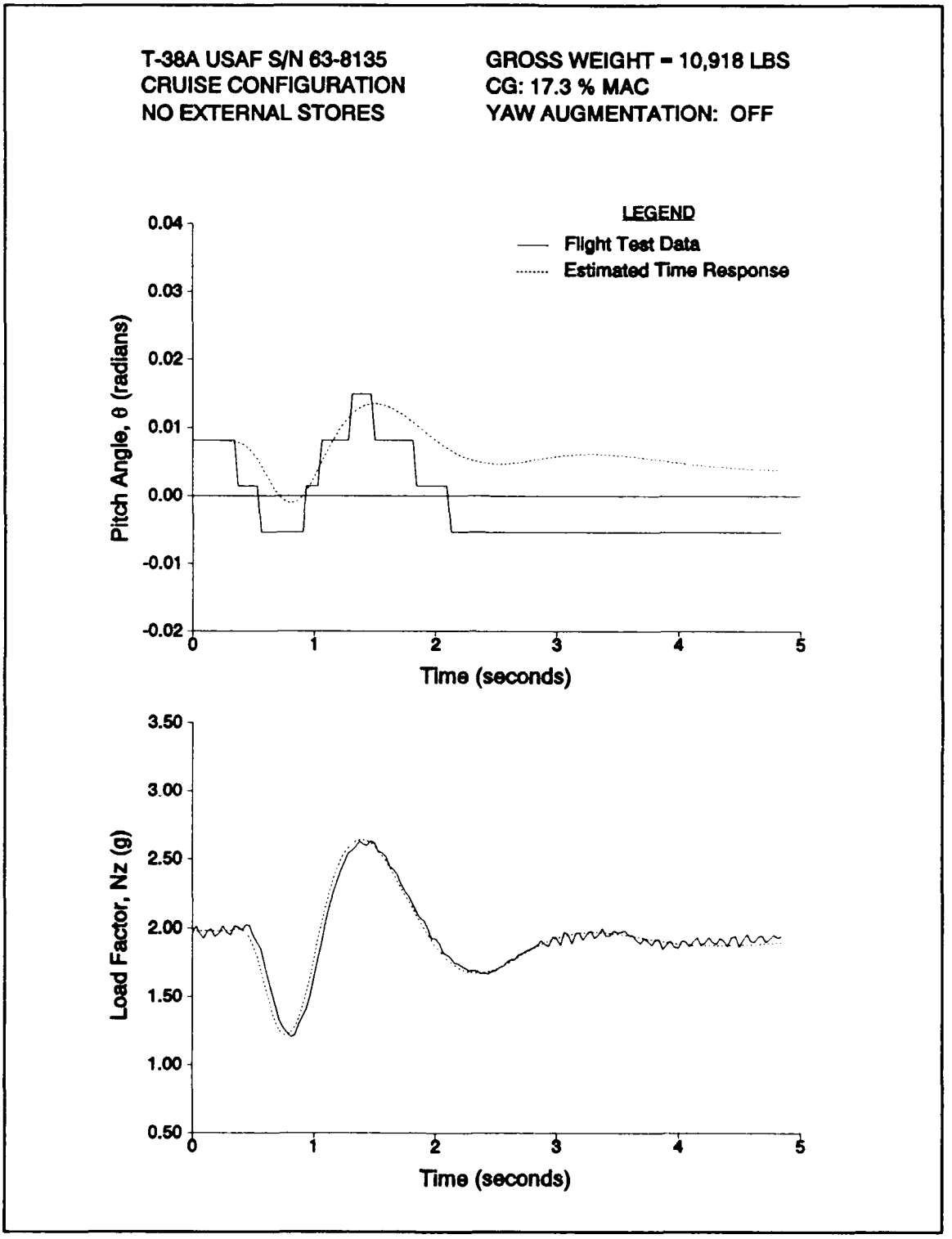

Figure B10. Time History Match, Test Point 4(b) (Sheet 2 of 2)

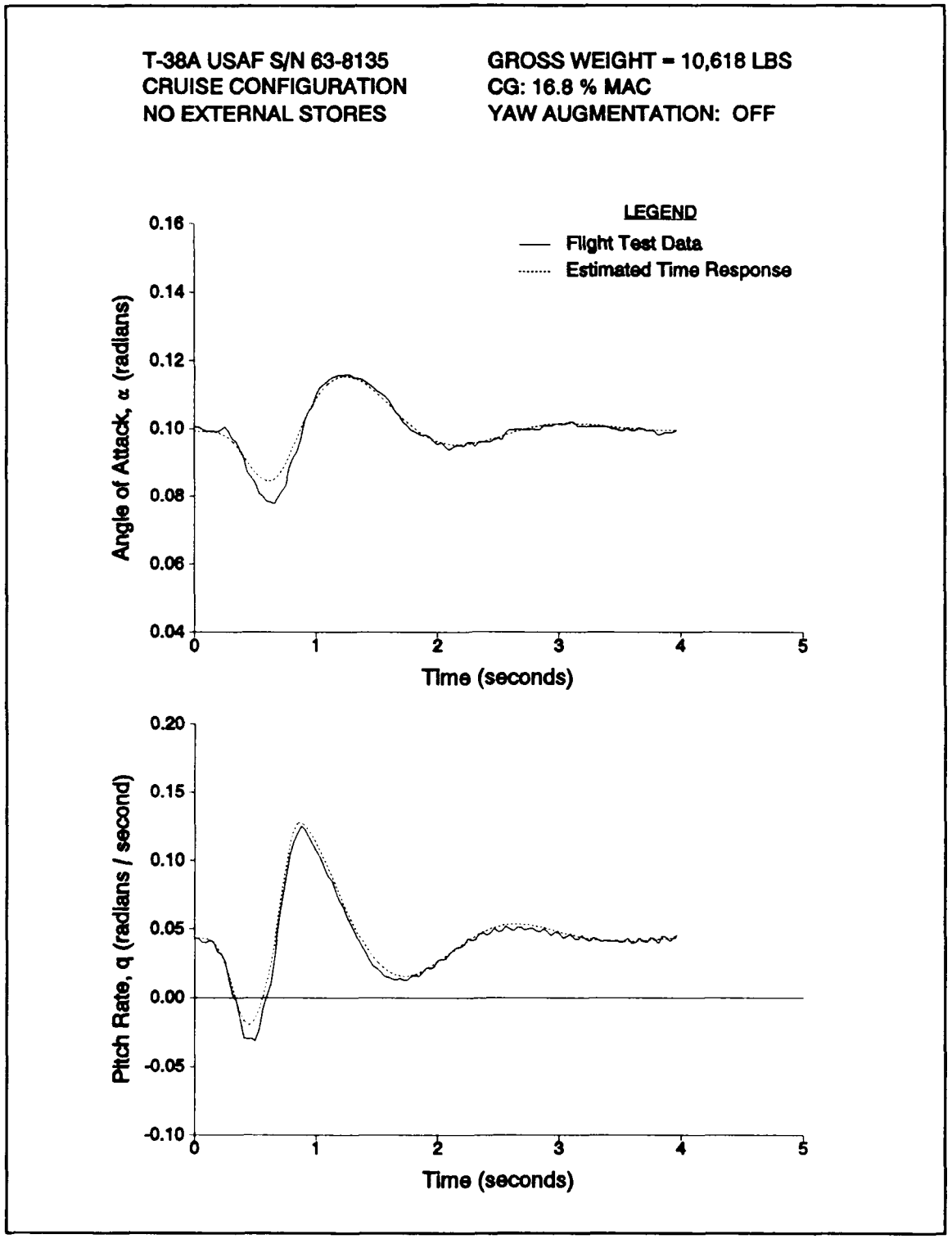

Figure B11. Time History Match, Test Point 4(c) (Sheet 1 of 2)

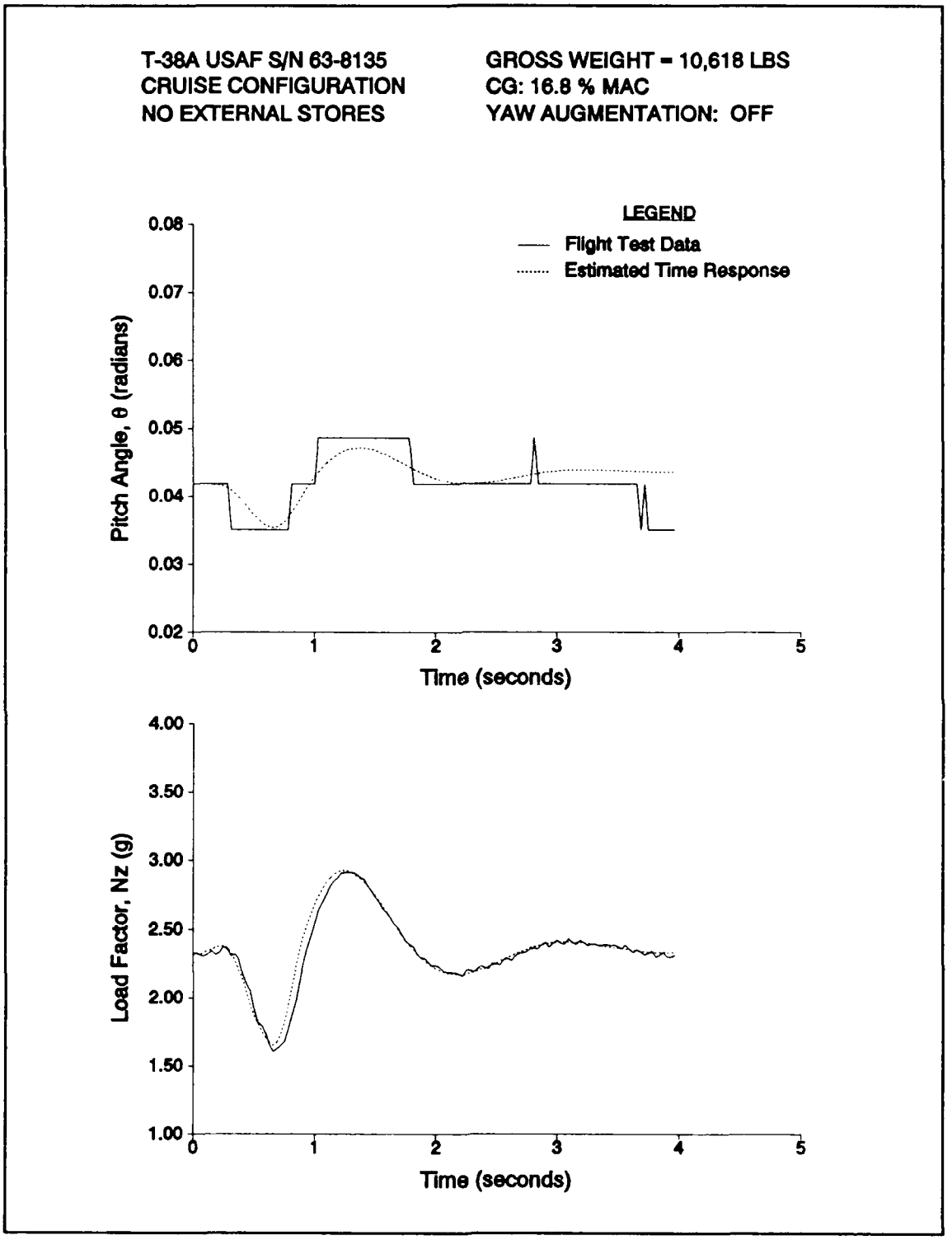

Figure B11. Time History Match, Test Point 4(c) (Sheet 2 of 2)

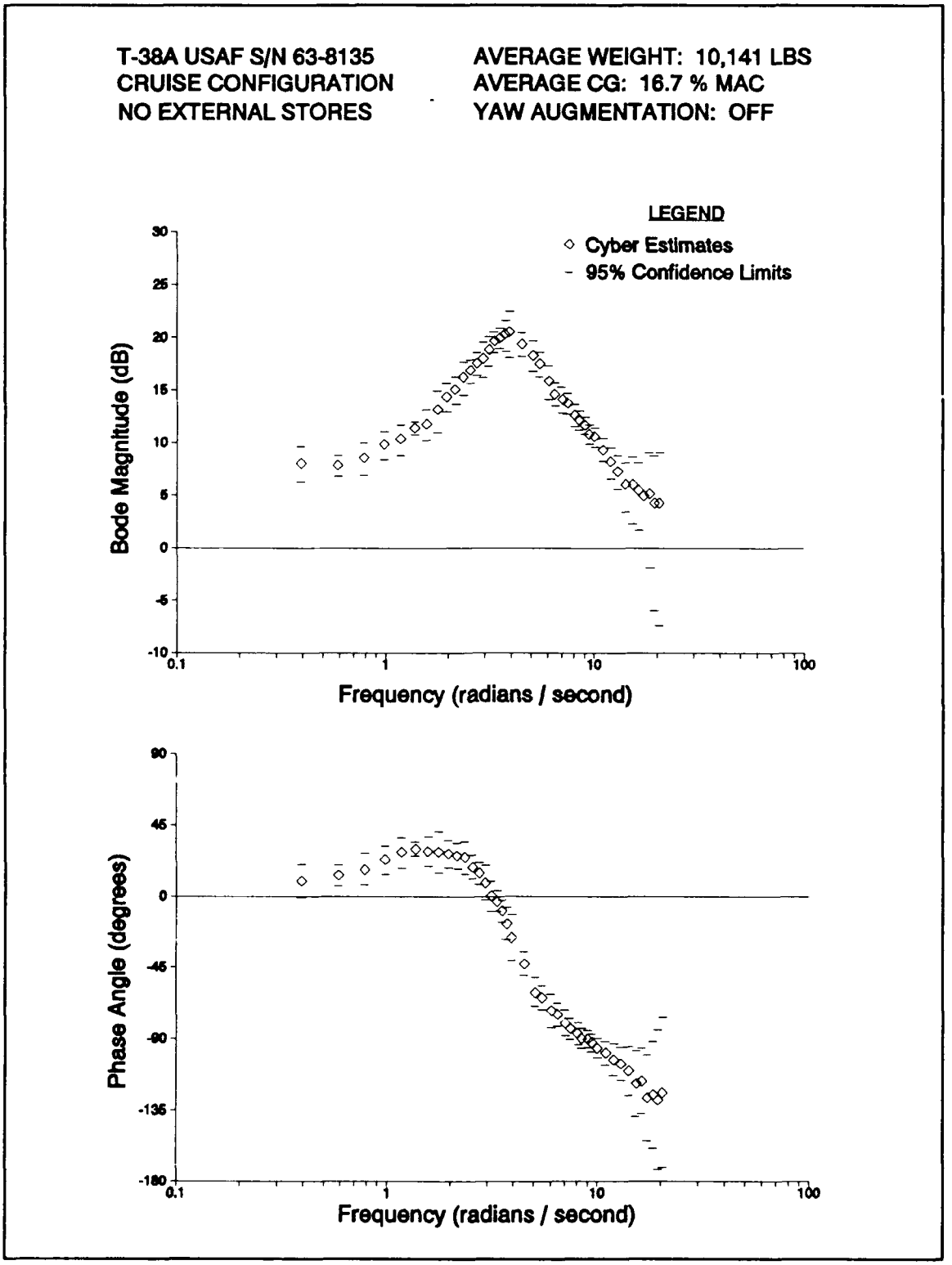

Figure B12. FRA Results, Test Point 2(a)

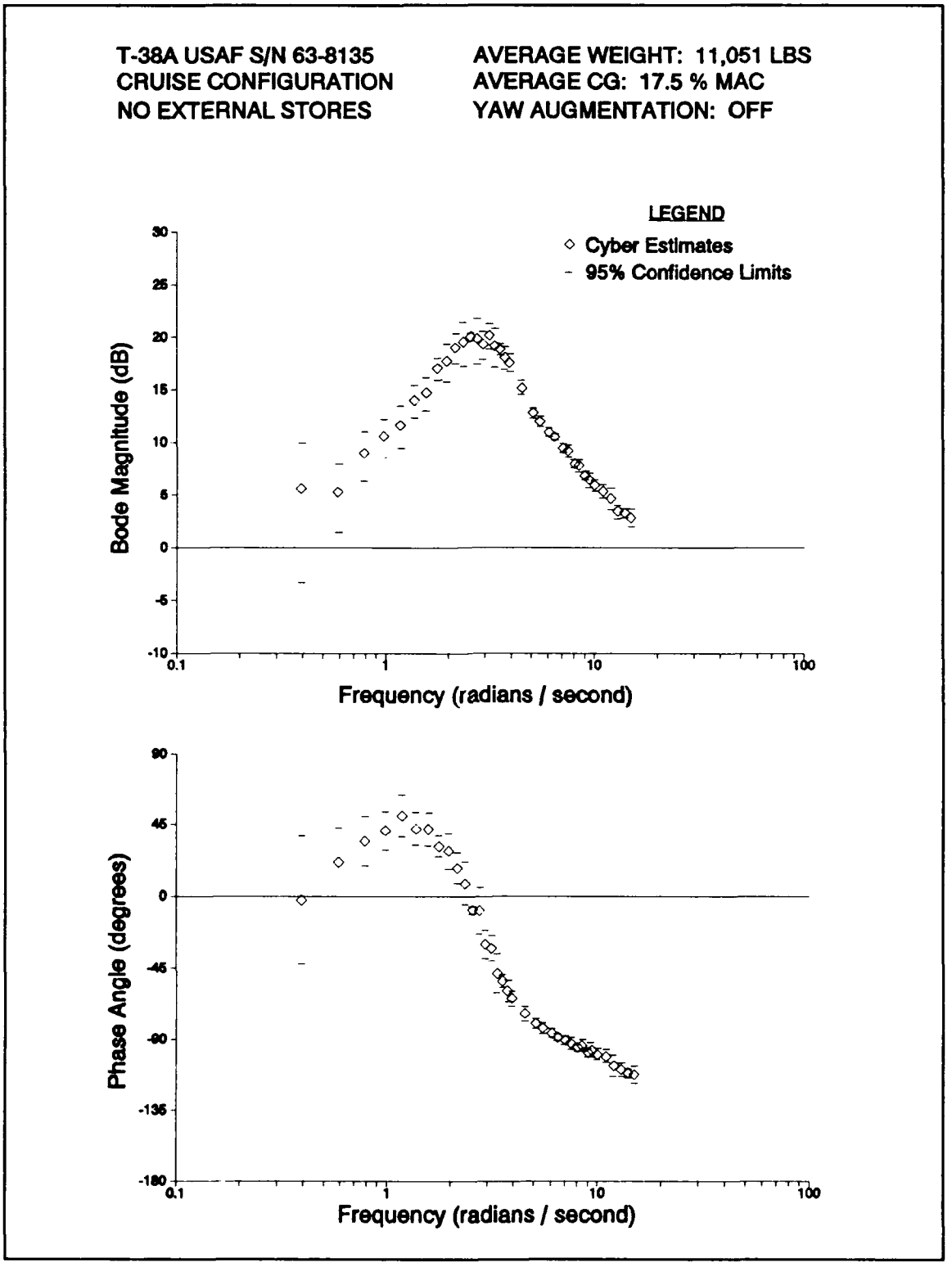

Figure B13. FRA Results, Test Point 3(a)

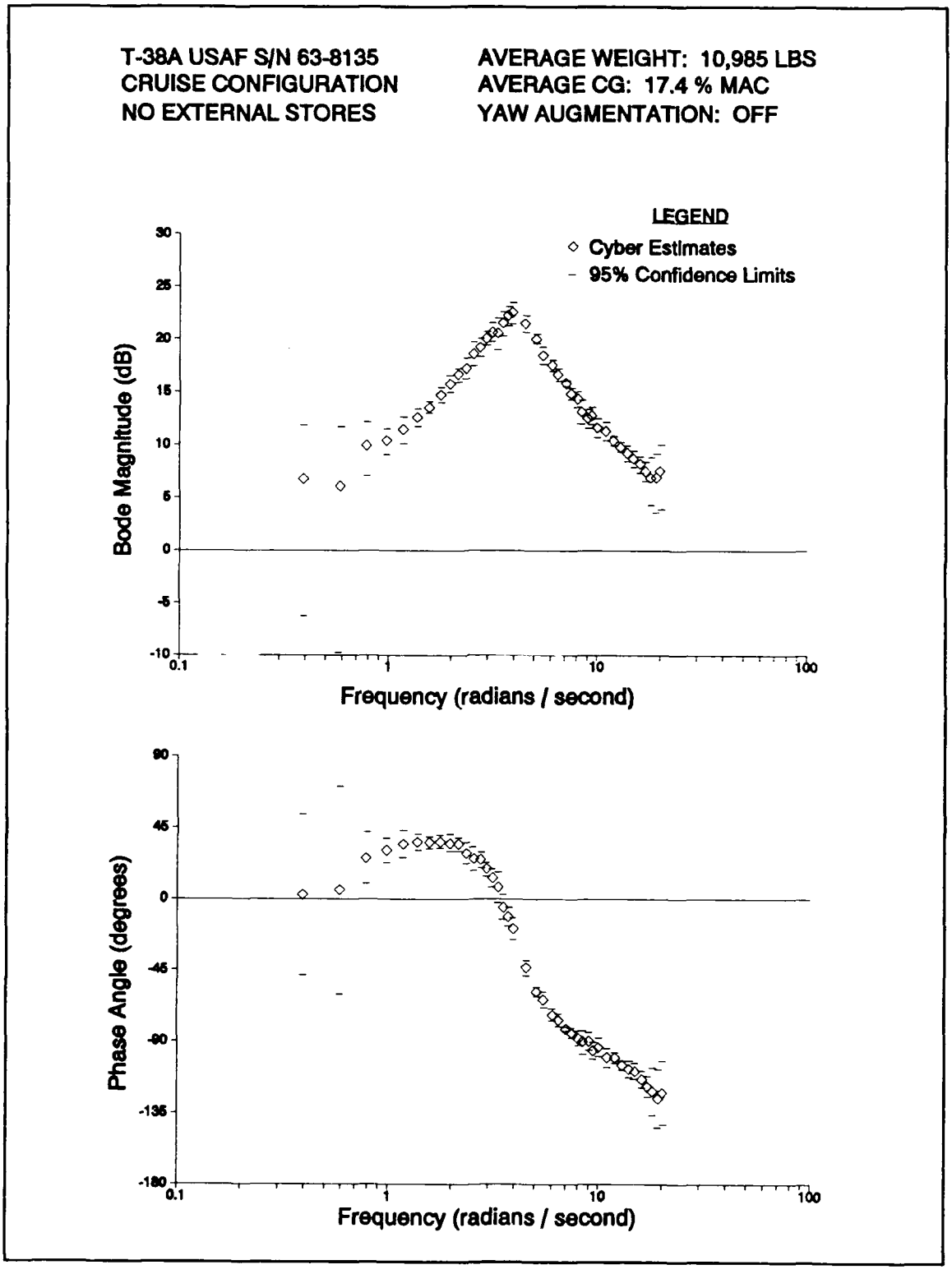

Figure B14. FRA Results, Test Point 4(a)

Bibliography

1. Flying Qualities Phase Textbook. USAF Test Pilot School. Edwards AFB, CA, September 1988.

2. Fromme, W. M. T-38 Static Stability and Basic Aerodynamic Data. Report No. NAI-58-578. Northrop Aircraft, Inc. Hawthorne, CA, October 1958.

3. Hasling, Robert F. and Siefke, Stanley P. Data Reduction on the Test Pilot School Data Reduction Computer System. USAF Test Pilot School. Edwards AFB, CA, July 1981.

4. Iliff, Kenneth W. and Maine, Richard E. "More Than You May Want to Know About Maximum Likelihood Estimation". AIAA Paper 84-2070, 1984.

5. Kitto, William G. and Berard, Thomas R. MMLE at AFFTC. 6521 Range Squadron/Computer Sciences Branch. Edwards AFB, CA, May 1984.

6. Little, John N. and Laub, Alan J. Control System Toolbox (for use with Matlab). South Natick, MA: The Math Works, Inc., August 27, 1986.

7. Maine, Richard E. and Iliff, Kenneth W. User's Manual for MMLE3, a General Fortran Program for Maximum Likelihood Parameter Estimation. NASA-TP-1563. Dryden Flight Research Center. Edwards, CA, November 1980.

8. Matlab for MS-DOS Personal Computers. South Natick, MA: The Math Works, Inc., February 3, 1989.

9. McRuer, Duane et al. Aircraft Dynamics and Automatic Control. Princeton, NJ: Princeton University Press, 1973.

10. Milne, Garth. State Space Identification Tool (for use with Matlab). Sherborn, MA: The Math Works, Inc., March 1988.

11. Nagy, Christopher J. A New Method for Test and Analysis of Dynamic Stability and Control. AFFTC-TD-75-4. Air Force Flight Test Center. Edwards AFB, CA, May 1976.

Bibliography (Concluded)

12. Ogata, Katsuhiko. Modern Control Engineering. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1970.

13. Olesen, T. J. Calculated Dead Weight Distribution and Moment of Inertia Report, T-38A. Report No. NAI-57-44. Northrop Aircraft, Inc. Hawthorne, CA, 1 March 1958.

14. Roskam, Jan. Airplane Flight Dynamics and Automatic Flight Controls. Ottawa, Kansas: Roskam Aviation and Engineering Corporation, 1979.

15. Shafer, Mary F. Flight Investigation of Various Inputs Intended for Parameter Estimation. NASA-TM-85901. Ames Research Center, Dryden Flight Research Facility. Edwards, CA, August 1984.

16. Uniform Flight Test Analysis System (UFTAS) Reference Manual, Version 3.0. 6521 Range Squadron/Computer Sciences Branch. Edwards AFB, CA, September 1989.

17. USAF Series T-38A Aircraft, T. O. 1-T-38A-1. 1 July 1987, Change 3, 1 May 1990.

Vita

Captain Mark S. Erickson was born on 17 January 1962 in Phoenix, Arizona. He graduated from high school in 1979 in Scottsdale, Arizona and attended the United States Air Force Academy, from which he received the degree of Bachelor of Science in Aeronautical Engineering in June 1983. Upon graduation, he was commissioned in the USAF and entered active duty. He worked as a flight test engineer at the F-15 Combined Test Force and at the 6512 Test Squadron, Air Force Flight Test Center, Edwards AFB, California until August 1988. He earned the degree of Master of Business Administration in Management from Golden Gate University in March 1987. He entered the School of Engineering, Air Force Institute of Technology, and completed academic requirements in December 1989. He graduated from the USAF Test Pilot School in December 1990, and was awarded the R. L. Jones Award as the top flight test engineer graduate. He was also nominated by the USAF Test Pilot School for the General James Ferguson Award for engineering achievement. Captain Erickson is currently assigned at the Air Force Flight Test Center, Edwards AFB, California.